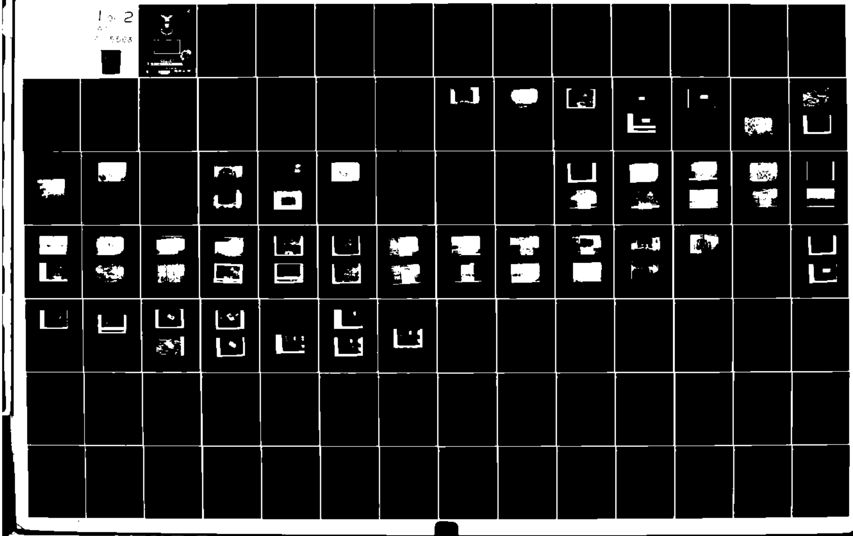
AD-A115 563 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 20/6
SCENE ANALYSIS - APPLICATION OF TWO-DIMENSIONAL NONLINEAR FILTE--ETC(U)
DEC 81 D R STROUP, T D DORSEY
UNCLASSIFIED AFIT/GE/GEO/EE/81D-57 NL

1 of 2
A115563



AFIT/GE/GEO/EE/81D-57

SCENE ANALYSIS - APPLICATION OF TWO-DIMENSIONAL NONLINEAR FILTERING FOR TARGET ENHANCEMENT AND RECOGNITION

THESIS

AFIT/GE/GEO/EE/81D-57 Darryl R. Stroup
Cpt. USAF
Thomas D. Dorsey
2nd Lt. USAF

Approved for public release; distribution unlimited.

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

AFIT/GE/GEO/EE/81D-57

SCENE ANALYSIS - APPLICATION OF
TWO-DIMENSIONAL NONLINEAR FILTERING
FOR TARGET ENHANCEMENT AND RECOGNITION

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by

Darryl R. Stroup
Cpt. USAF
Graduate Electrical Engineering

Thomas D. Dorsey
2nd Lt. USAF
Graduate Electro-optics

December 1981

Approved for public release; distribution unlimited.

Acknowledgements

We gratefully acknowledge the advice, encouragement and discussions with our thesis advisor, Dr. Matthew Kabrisky, Professor of Electrical Engineering. We also wish to acknowledge the help of Captain Larry Kizer in our successful quest to understand the Digital Signal Processing Equipment. We also gratefully acknowledge the assistance of our co-worker in software development, 2nd Lt. Robin Simmons.

Contents

Page

Contents

List of Figures

List of Figures

Figure | Page

List of Figures

Abstract

A nonlinear scene analysis algorithm is studied for complex scenes containing realistic targets and background clutter. Infrared and visible spectrum light images are processed. The algorithm combines the Fourier transform phase array of a scene with the Fourier transform magnitude of a template to create a new image. Clutter reduction ability and target recognition capability are examined in detail.

SCENE ANALYSIS - APPLICATION OF
TWO DIMENSIONAL NONLINEAR FILTERING
FOR TARGET ENHANCEMENT AND RE౿OGNITION

I Introduction

Machine detection and recognition of a target in a complex or cluttered scene is a major concern in advanced military technology (Ref 1,2). Areas of application are analysis of reconnaissance photos, weapon delivery display systems and "smart" munitions.

Background

A target acquisition process can be viewed as consisting of three main procedures. The first procedure is the development of a representation of a scene that can be used as input data. The second procedure takes the input data and extracts characteristic features and attributes. The final procedure is the classification and identification of the extracted features (Ref 3). This paper assumes that the first procedure is well defined and that the input data used are images formed from light in the visible spectrum, or images formed by energy from the infrared spectrum.

The major problem of detection and recognition of targets in cluttered scenes is extracting the characteristic features needed to classify the target. The target can be embedded in a virtually infinite variety of backgrounds with many of these backgrounds having the same general features and characteristics as the target. This causes the final procedure of target classification and identification to be extremely difficult. No general solution to this problem is known.

In 1980, Moshe Horev developed a picture correlation algorithm to detect and identify an object embedded in a cluttered scene. This detection could be accomplished without prior knowledge of the objects' size, orientation or location (Ref 4). This algorithm combined the feature extraction process and the identification process in a template matching scheme that is accomplished in two stages:

1. The first stage begins by a series of transformations that places the template and scene into a special "correlation" plane. A correlation is performed. The location of the peak value of the correlation provides the angle of rotation and the difference in scale between the target and the template. The template is then modified so that it is at the same scale and angle of rotation as the target.
2. In the second stage the phase angle array from the Fourier transform of the image is combined with the magnitude array of the Fourier transform of the modified template. Throughout the remainder of this thesis this process will be referred to as the PIMT (Phase of Image, Magnitude of Template) process. An inverse Fourier transform of the combined array is computed. The result is a modified image, in which the target is enhanced while the clutter is suppressed.

Horev used the algorithm to successfully identify targets in two photographs with each photograph having a different background clutter. The significance of these successes is the apparent ability of the algorithm to suppress background clutter. In a separate study (Ref 5), infrared pictures were processed using the algorithm. The results were unsuccessful. No detailed study has been made of the reasons of the success or failure of the algorithm.

Problem

The purpose of this thesis is to investigate the effectiveness of the second stage of the Horev algorithm in suppressing background clutter.

Equipment, Software and Data Base

The image processing for this project was accomplished in the digital signal processing laboratory at the Air Force Institute of Technology.

Visible spectrum images were developed in the laboratory using a system consisting of a CVC-1 Vidicon camera, video moniter and a NOVA 2/10 minicomputer interfaced with a Cromemco microcomputer. The images produced were 256 X 256 pixel arrays, with each pixel grey scale value ranging from 0 to 15.

The infrared data base consists of 945 digitized infrared images provided by the Air Force Armament Laboratory, Elgin AFB, Florida. The images are in TABILS format on magnetic tape. Each image is a 100 X 100 pixel array, with each pixel grey scale value ranging from 0 to 1023. Since this thesis is concerned only with the processing of the final digitized image, a detailed description of the data base is not provided. However, a detailed description of the TABILS format data base can be obtained in chapter II of reference 5.

All computations required for image processing were accomplished on a Data General Eclipse S/250 that shares disk memory with the NOVA/Cromemco system.

The photographs contained in this thesis were obtained from the NOVA/Cromemco system video moniter. A complete description of the AFIT image processing facility hardware and software is contained in reference 6. Software unique to this thesis is contained in the appendix.

Scope

The PIMT process was used on filtered and unfiltered images composed of different scenes and targets. A threshold operation was performed on the original image and the PIMT image. If the clutter was reduced and the target enhanced in the PIMT image, when compared to the original image, the process was considered a success. This study does not attempt to develop a rigorous mathematical analysis of the nonlinear PIMT process. However, a theory is developed and presented based on a study of the results of the process.

Sequence of Presentation

This study begins with a discussion of the PIMT process theory and implementation. Chapter III discusses the processing used to create the scene and template images used. Chapter IV verifies that the process of Chapter II (PIMT Process) obtains the same results as those obtained by the second stage of the Horev algorithm. Chapters V and VI examine the effect of filtering on the PIMT process. Chapter VII examines the effect of size and rotation variations and Chapter VIII examines the discrimination ability of the process.

II PIMT Process

Theory

The PIMT process is the combination of the phase array from the Fourier transform of the scene combined with the magnitude array of the Fourier transform of the template. Let the intensity functions of the template image and scene image be designated as t(x,y) and s(x,y), respectively. Let the 2-D DFT be designated as in equations 1 and 2.

$$F\{t(x,y)\} = T(\zeta,\eta) \qquad (1)$$

$$F\{s(x,y)\} = S(\zeta,\eta) \qquad (2)$$

The Fourier transform is a complex function and can be represented as:

$$T(\zeta,\eta) = R\ (\zeta,\eta) + j\ I\ (\zeta,\eta) \qquad (3)$$

$$S(\zeta,\eta) = R\ (\zeta,\eta) = j\ I\ (\zeta,\eta) \qquad (4)$$

where $R(\zeta,\eta)$ and $I(\zeta,\eta)$ are the real and imaginary components of the transformation. Using Euler's equation, $T(\zeta,\eta)$ and $S(\zeta,\eta)$ can be represented in terms of their magnitude and phase spectrum as shown below.

$$T(\zeta,\eta) = |T(\zeta,\eta)| \exp[j\phi t(\zeta,\eta)] \qquad (5)$$

$$S(\zeta,\eta) = |S(\zeta,\eta)| \exp[j\phi s(\zeta,\eta)] \qquad (6)$$

It should be noted that $T(\zeta,\eta)$ and $S(\zeta,\eta)$ are each represented as a discrete 256 X 256 complex array. Every point in that array has a magnitude and phase associated with that point. The result of the PIMT process can be designated as shown in equation 7.

$$P(\zeta,\eta) = |T(\zeta,\eta)| \exp[j\phi_s(\zeta,\eta)] \qquad (7)$$

Looking at a particular point ζ=a, η=b then

$$T(a,b) = |T(a,b)| \exp[j\phi_t(a,b)] \qquad (8)$$

$$S(a,b) = |S(a,b)| \exp[j\phi_s(a,b)] \qquad (9)$$

$$P(a,b) = |T(a,b)| \exp[j\phi_s(a,b)] \qquad (10)$$

The point (a,b) in the Fourier (spatial frequency) domain represents a location on the spatial frequency plane. If the scene consists of background clutter plus target, then the phase component of the frequency spectrum is

$$\phi_s(a,b) = \phi_t(a,b) + \theta(a,b) \qquad (11)$$

where $\theta(a,b)$ is the phase change due to the background clutter or noise present in the scene. If the noise component, $\theta(a,b)$ is zero or very small for all (a,b), then

$$P(a,b) \simeq T(a,b) \qquad (12)$$

and the image of the target would be essentially unaffected in the PIMT image. If, however, the term $\theta(a,b) \neq 0$ for all (a,b), then the target information will be distorted in the PIMT image. Similarly, if the template magnitude, $|T(a,b)|$, is less than the scene magnitude, $|S(a,b)|$, for any (a,b), then the target energy/clutter energy ratio, or the signal to noise ratio should be increased causing a reduction in the background clutter. Thus, the PIMT process can be seen to be optimized if the spatial frequencies of the clutter differ significantly from the frequencies of the target making the PIMT process both scene and target dependent. This will be demonstrated in the following sections.

Implementation

The PIMT process was implemented as shown in the flow chart of Figure 1.

The first step involved computing two dimensional fast Fourier transforms (2-D DFT) for both the template and scene images. If filtering was used, it occurred in the spatial frequency domain as shown in the processing steps designated A and B of Figure 1. The PIMT process was

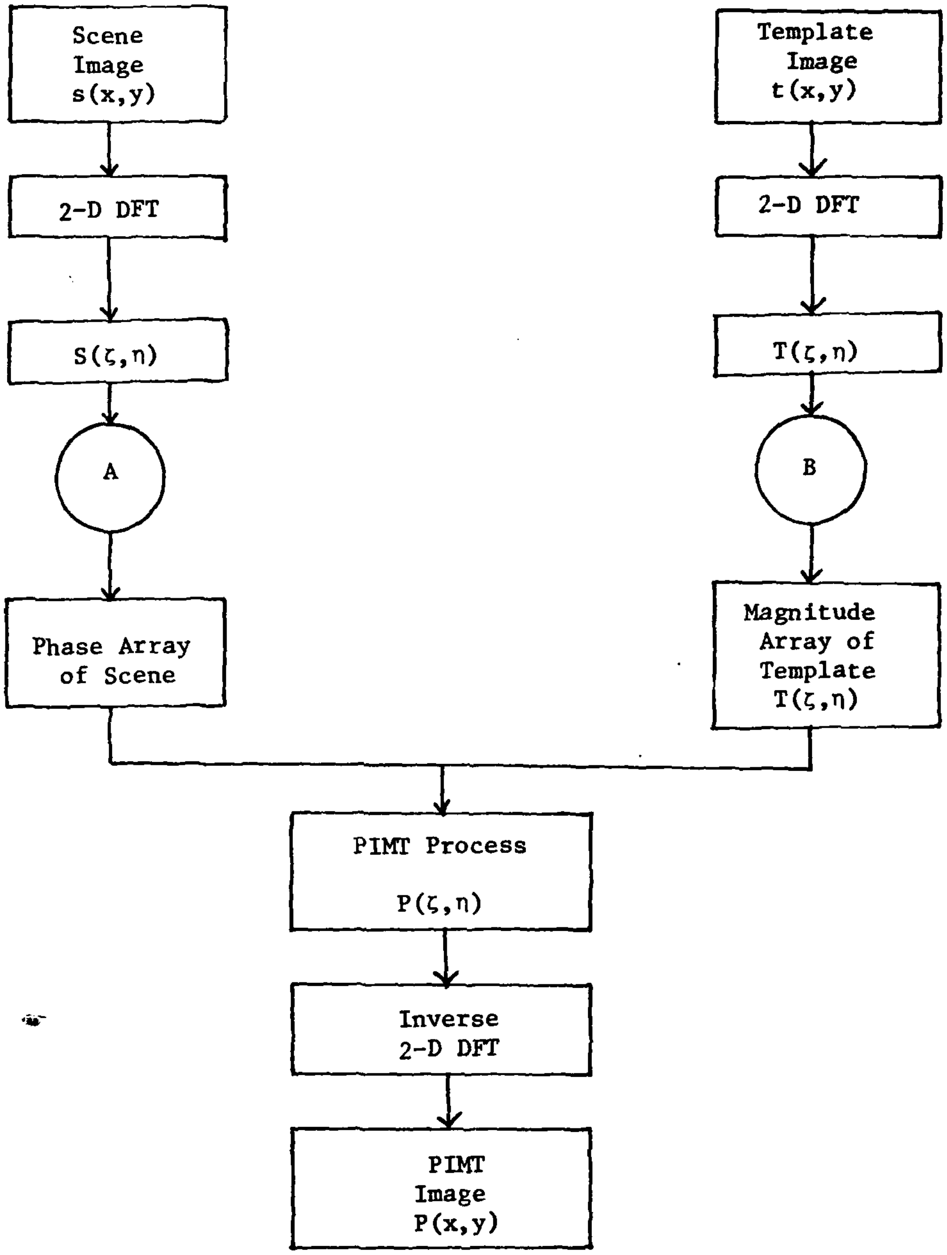


Figure 1. PIMT Process

then performed to combine the Fourier transform phase array of the image with the Fourier transform magnitude array of the template. An inverse 2-D DFT was then performed on the result of the PIMT process to form the PIMT image. For the remainder of this thesis, the forward and inverse 2-D DFT's will be considered an integral part of what will be referred to as the PIMT process.

III Image Processing

The image processing used, in addition to the PIMT process, were the processes used to create the images and those used to filter the created images. This chapter will contain a discussion of the procedures used to create the scene and template images. A discussion of the filtering processes will be presented in Chapter VI.

Infrared Spectrum

All processing of the infrared images (including filtering) utilized the full 1024 grey-scale levels provided by the data base. The photographs shown in this section and the remainder of this thesis were obtained from the NOVA/Cromemco system video monitor. Since this system only uses a 16 level grey scale, a linear scaling transform (Ref 7:161) was performed as a final step to allow display of the infrared images.

A template of a tank was created from the first IR scene of the data base. This scene is shown in Figure 2.

The procedure used to create the template was an edge search operation as shown in Figure 3. The operation starts at the right edge of the scene (pixel A_0) and computes the difference between two adjacent pixels in the row ($A_N - A_{N-1}$). If the difference is below or equal to a certain threshold (T), the difference is then calculated and tested for the next two pixels. This continues until the difference exceeds the threshold value. When this occurs, all of the pixels in the row from A_0 to A_N are set equal to zero completing the operation. The same procedure was applied to the top, left, and bottom edges. The resulting template is shown in Figure 4. This procedure worked well in the scene

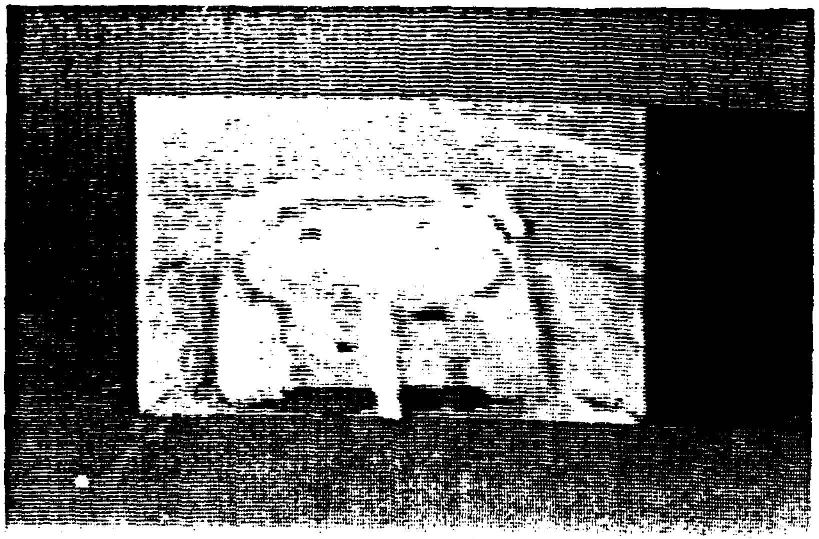


Figure 2. Scene Used to Create the Infrared Template

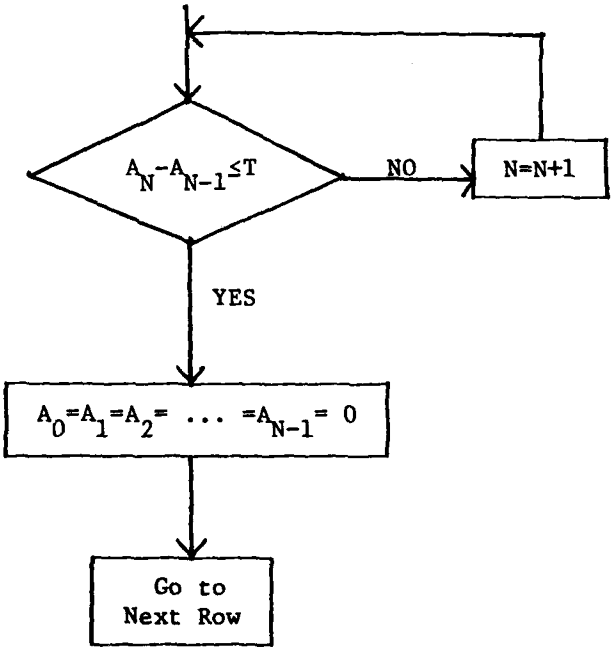


Figure 3. Edge Search Operation

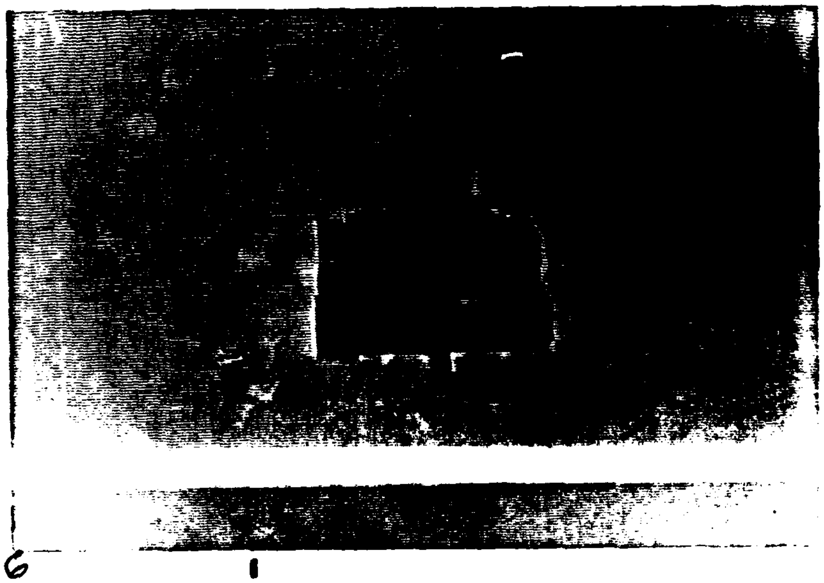


Figure 4. Infrared Tank Template

of Figure 2 because there were no high energy spots (hotspots) in the background and the tank was significantly "hotter" than the rest of the scene. The procedure is not a general one and it is likely that it would fail when applied to most scenes.

A three by three image array was created as a master scene to test the PIMT process. The nine infrared images, including the scene of Figure 2, were normalized to insure that no particular image would be "hotter" than any other particular image used in the master scene. The nine images were combined to produce the master scene shown in Figure 5.

Visible Spectrum

Visible spectrum images were created in the laboratory from black and white photographs. Each photograph was digitized into a 256 X 256 pixel array. Random noise introduced during digitization was reduced

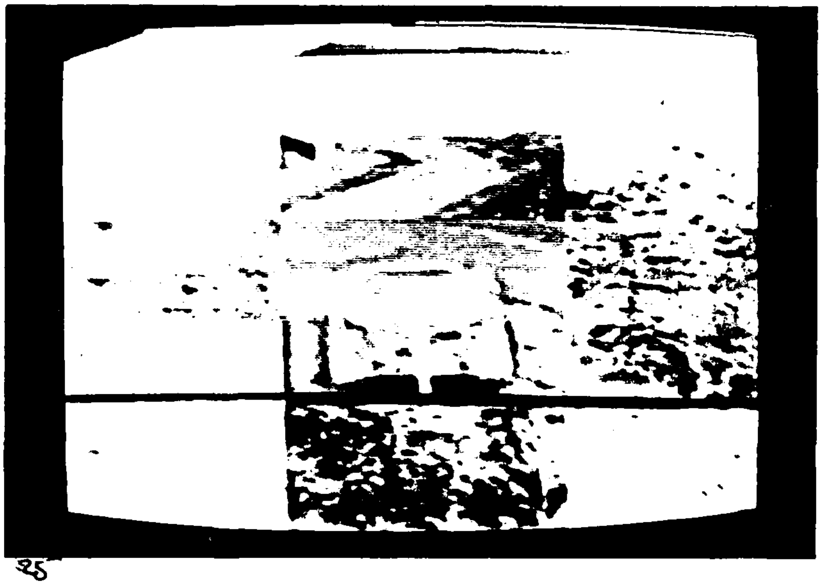


Figure 5. Master Scene of Infrared Images

using frame-to-frame averaging (Ref 7:206). The averaged image was then tested for the minimum and maximum grey scale level. A linear scaling transform (Ref 7:161) was used, when necessary, to fully utilize the 16 grey levels available with the NOVA/Cromemco system.

Templates were created by first segmenting the desired object into a rectangular window surrounded by a zero grey-scale level background. A "trial-by-error" thresholding was then done on the window until only the desired template remained. Figure 6 shows a black and white photograph prior to digitization. Figure 7 shows the result of the digitization and noise reduction process and Figure 8 is a completed template.

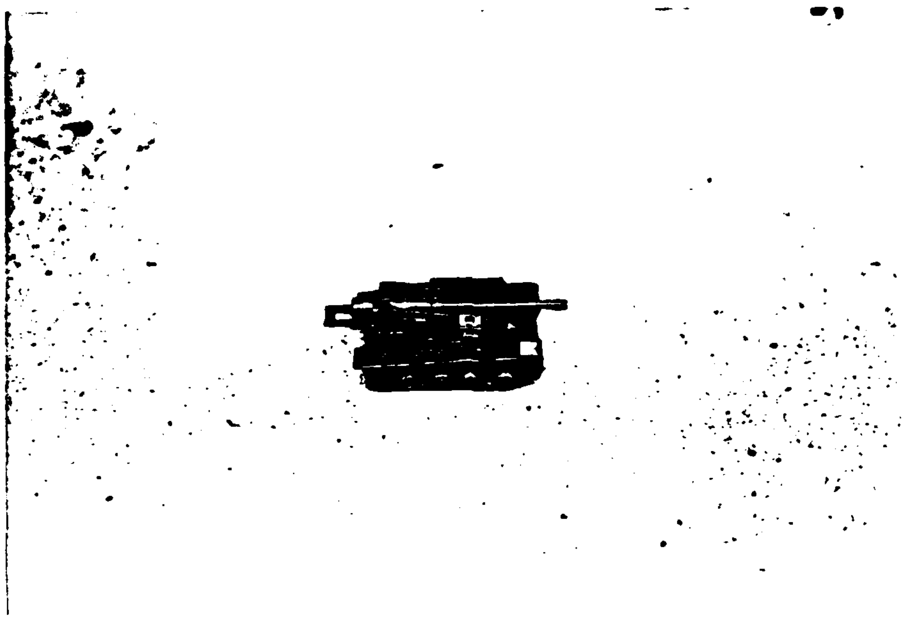


Figure 6. Unprocessed Photograph

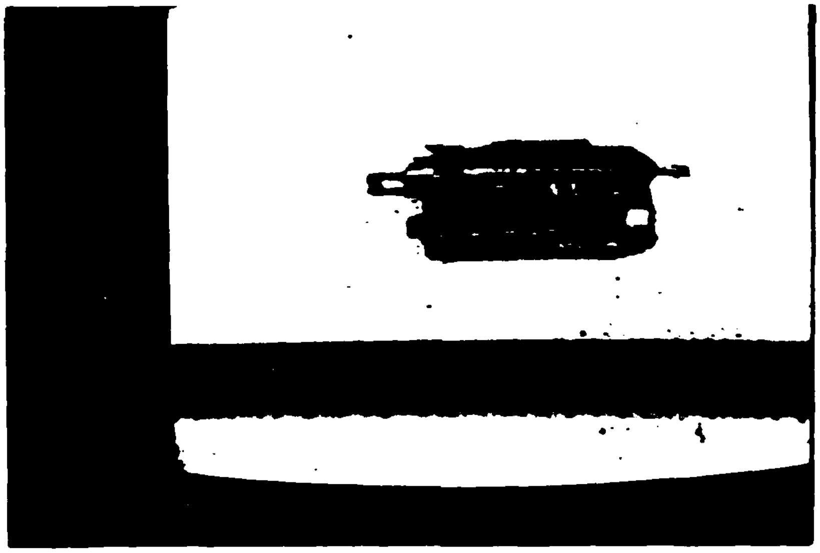


Figure 7. Digitized and Noise Reduced Image

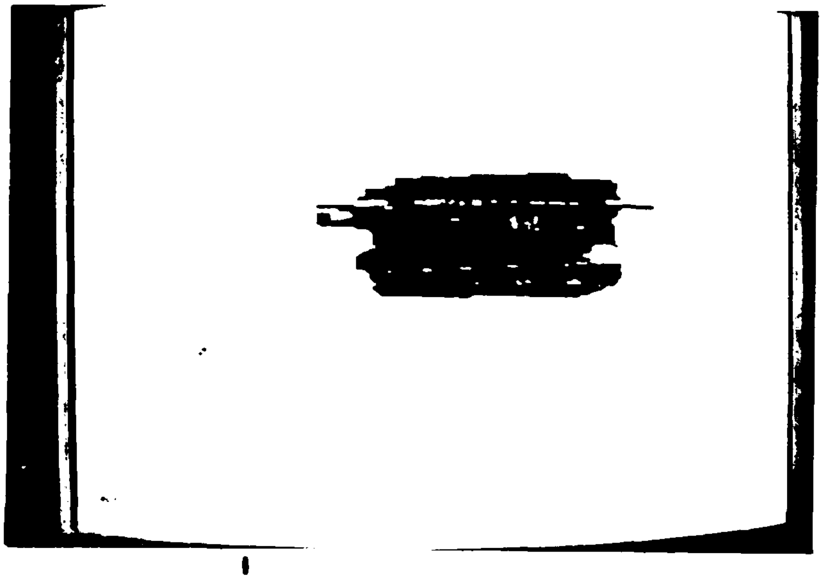


Figure 8. Final Template

The template could be moved to any location within the array and combined with any background scene. This process guaranteed that the target in the scene would be the same scale and rotation as the template used, in accordance with the original assumption.

IV Verification of the PIMT Process

The stated purpose of this thesis is to investigate the effectiveness of the second stage of the Horev algorithm in suppressing background clutter. Two samples of results are presented in this chapter to demonstrate that background clutter reduction produced by the PIMT process is similar to that obtained using the Horev algorithm. Since the scenes and templates used by Horev were not available for this test, success is based solely on the similarity of the clutter reduction obtained.

Procedure

PIMT processing was the only procedure used to obtain the results contained in this chapter. The scenes used are shown in Figures 9A and 9B. The template used for both scenes is shown in Figure 10.

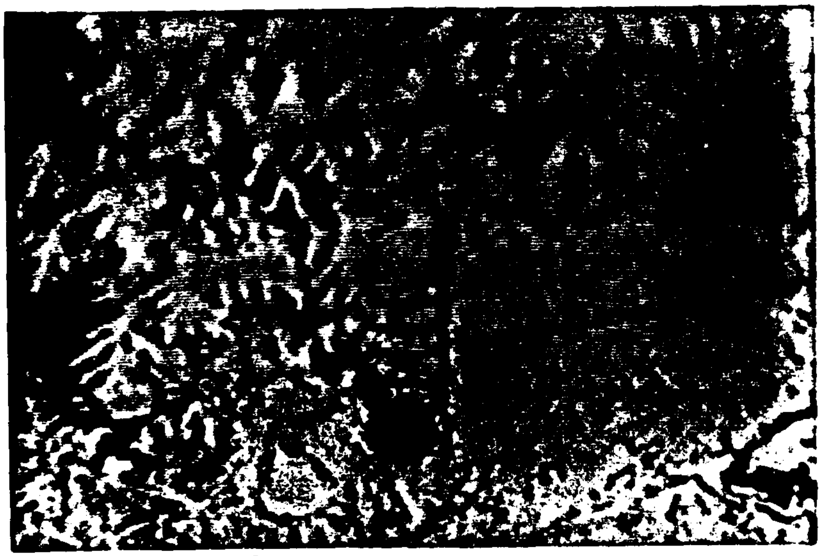


Figure 9A. First Scene Image

Figure 9B. Second Scene Image

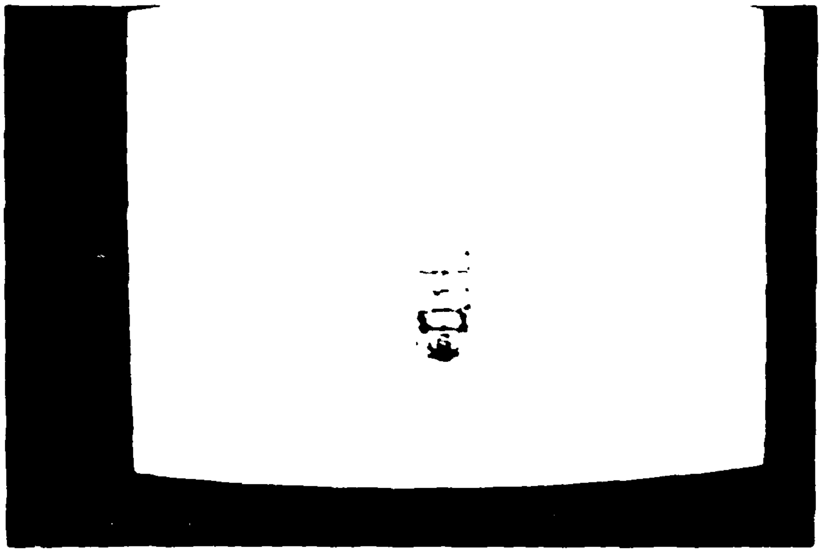


Figure 10. Template

Results

The final PIMT images are shown in Figures 11A and 11B. It can be seen by comparing Figures 9A and 11A and Figures 9B and 11B that a significant reduction in background clutter occurs as a result of the PIMT process. This does demonstrate that the PIMT process reduces clutter similar to the clutter reduction obtained by the Horev algorithm. Further analysis of the images shown in Figures 11A and 11B will be discussed in chapter IX.

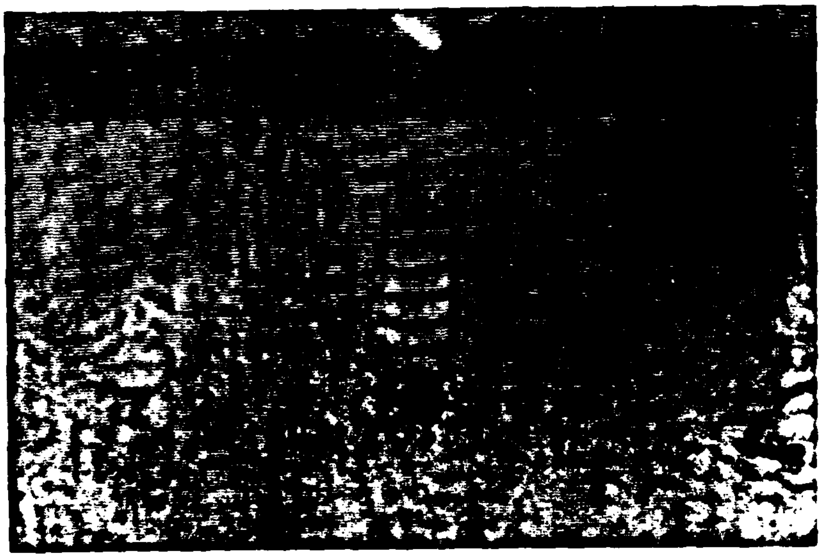


Figure 11A. PIMT Image of the Scene of Figure 9A

Figure 11B. PIMT Image of the Scene of Figure 9B

V Small Window Processing

Horev hypothesized that the PIMT process works best when a small window is used (Ref 4:87). The theory is that there will be a low clutter-energy to target-energy ratio, since most of the scene will consist of the target. The infrared images were already in a small window format and, therefore, used to test Horev's hypothesis.

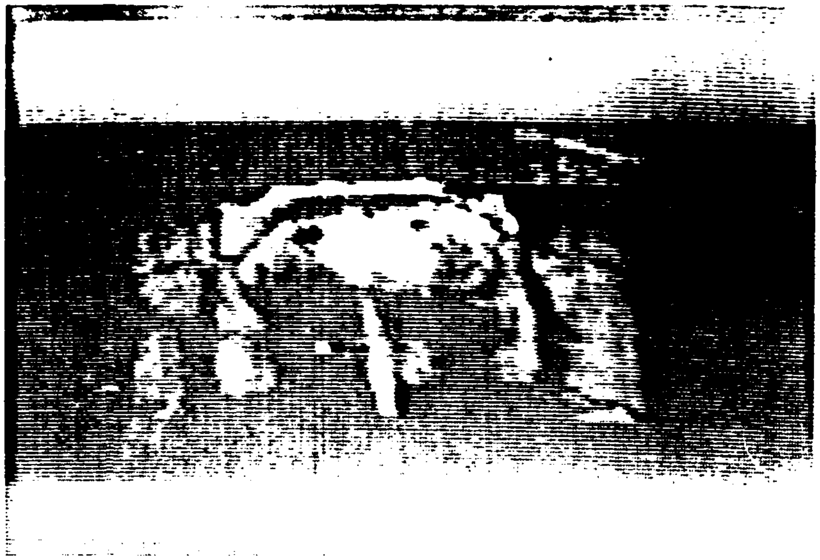
Procedure

The PIMT images were created using the template shown in Figure 4 and scenes shown in Figures 2, 13A and 14A. Correlations were performed between the template image and both the PIMT image and the original scene image. The two correlations were then compared to see if the PIMT processing resulted in a higher correlation between the processed scene and the template.

Results

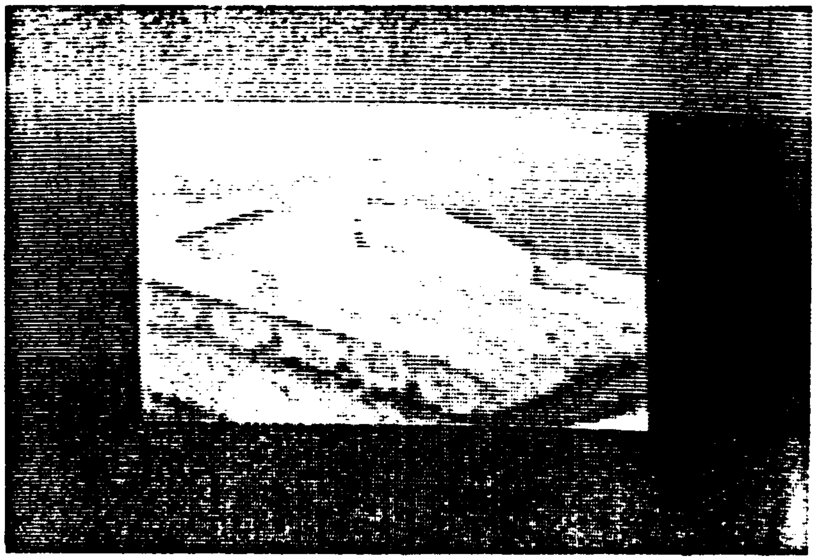
Figure 12 is the PIMT image produced using the template of Figure 4 and the scene of Figure 2. Figures 13A and 14A are original scene images and Figures 13B and 14B are the resulting PIMT images respectively.

It should be noted that even with a small window there is a tendency for the PIMT image to be merely a distorted reproduction of the original scene. A comparison of the correlations between the PIMT images and the template, and of the correlations between the original scenes and the template shows that the correlations using the original scenes worked better in target detection. In either case, there was no significant difference between the two operations.



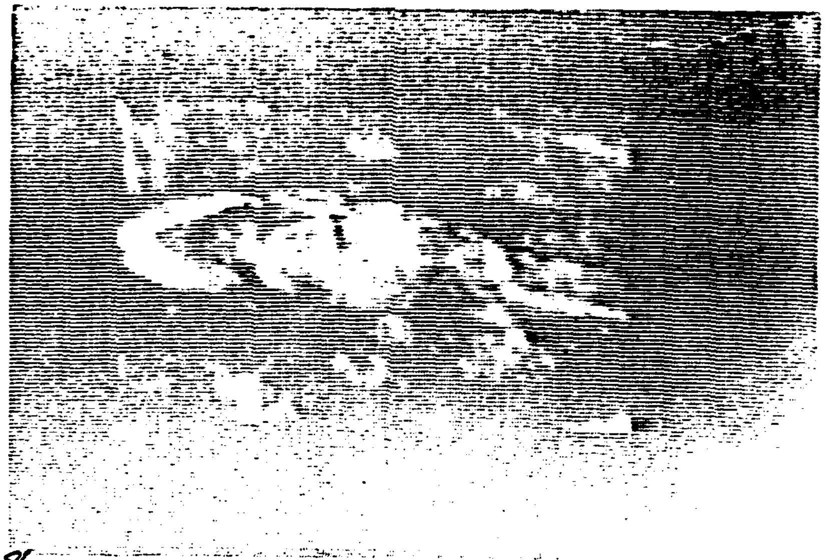
3

Figure 12. PIMT Image of Scene in Figure 2



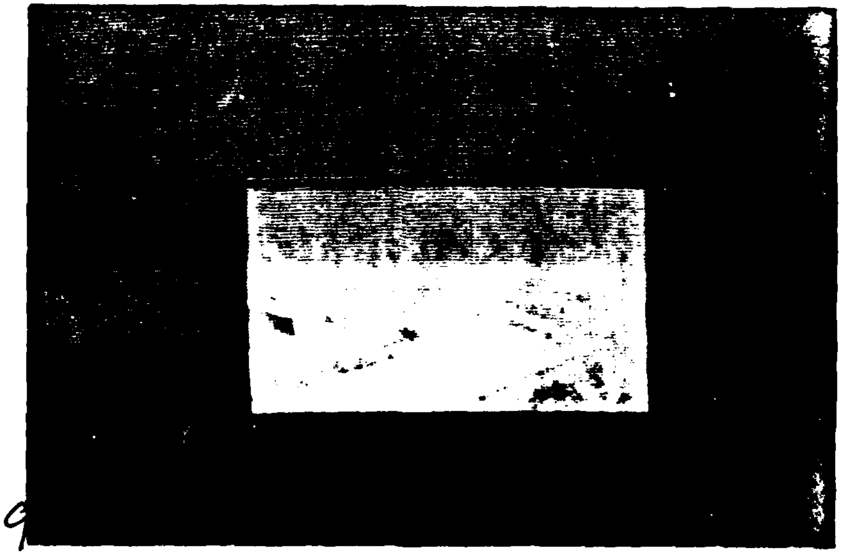
7

Figure 13A. Scene with Target Rotated 135 Degrees



8

Figure 13B. PIMT Image of Scene in Figure 13A



9

Figure 14A. Scene Containing No Target

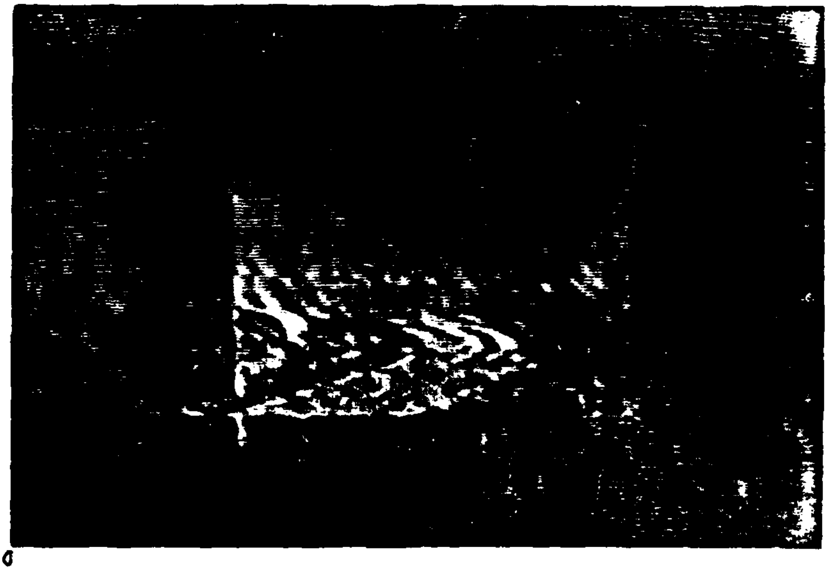


Figure 14B. PIMT Image of Scene in Figure 14A

VI Filtering

Various filters were used in conjunction with the PIMT process. All filtering was accomplished in the spatial frequency domain as shown in the processing steps designated as A and B in Figure 1. This chapter will briefly describe the filters used on the scene images and discuss the results on the PIMT image.

Filters

The filters used, as described in this section, are a function of a radial frequency f. This radial frequency describes a radius in the frequency domain starting from the centered DC component to f. The following filters were used:

$$H(f) = 1 \quad \text{(no filtering)} \tag{13}$$

$$H(f) = f \tag{14}$$

$$H(f) = 1/f \tag{15}$$

$$H(f) = f^2 \tag{16}$$

$$H(f) = \begin{cases} f & (f \leq 12) \\ 1/f^2 & \end{cases} \tag{17}$$

$$H(f) = \begin{cases} 1 & (f_1 \leq f \leq f_2) \\ 0 & \text{(0 elsewhere)} \end{cases} \tag{18}$$

The filters described in Equations 14 and 16 were used to duplicate the effect of the transformations used by Horev in his first stage of processing (Ref 4:40). The filter of equation 15 is essentially a low-pass filter and was used to investigate the effects of low-pass filtering on the PIMT process. The band-pass filter shown in equation 18 was used to accomplish a frequency analysis of the infrared image of

Figure 5. It was used to find a frequency range in which the PIMT results would be more successful. Photographs of these filtered scenes and resulting PIMT image are included at the end of this chapter.

Three other filter type operations were used in an attempt to improve the PIMT process. These operations compared the scene with template to try to emphasize those frequency components of the template which were much stronger than corresponding frequency components of the image. A new template, T'(ζ,η) was created as shown in Equations 19 through 21 below.

$$|T'(\zeta,\eta)| = \begin{cases} |T(\zeta,\eta)|-|S(\zeta,\eta)| & \text{if } |T(\zeta,\eta)|-|S(\zeta,\eta)|>0 \\ 0 \text{ Elsewhere} & \end{cases} \qquad (19)$$

$$|T'(\zeta,\eta)| = \frac{|T(\zeta,\eta)|}{|S(\zeta,\eta)|} \qquad (20)$$

$$|T'(\zeta,\eta)| = \frac{|T(\zeta,\eta)|^2}{|T(\zeta,\eta)|-|S(\zeta,\eta)|} \qquad (21)$$

Equation 21 was used because Horev predicted that it would automatically yield "the 'best' set of features" (Ref 4:102).

Results

Figures 15 through 47 show the results of the various filters on infrared images. It can be seen that none of the filters caused significant improvement in the PIMT process. Similar results were obtained using imag from the visible spectrum.

These results are best interpreted using Equation 7. In the analysis in Chapter II, it was proposed that the clutter reduction largely came about when, for any particular ζ=a and η=b, |T(a,b)| was very small or equal to zero. This causes the corresponding phase component $\phi_s(a,b)$ to be ineffective. This would reduce the non-template frequency components of the scene image. While the corresponding template frequency components, i.e. the target, would be unaffected. Since the filtering caused no improvement in the PIMT process there are at least three possible conclusions. One, the right filter function had not been applied. Two, the scene and the template have similar frequency magnitude spectrums. Three, the PIMT process is independent of the template frequency magnitude spectrum.

Reference 8 and Chapter VIII of this thesis present experimental evidence that for target identification, the PIMT process is independent of the template magnitude. Although no detailed comparison was made, examination of the 2-D DFTs of the scenes and the templates have shown that both have similar spectrums. Consistent failure of spectral filtering techniques when applied to complex scenes was the basis for assuming similar spectrums at the start of this research. Given the evidence supporting the last two conclusions it is the authors' opinion that there is no "right" filter.

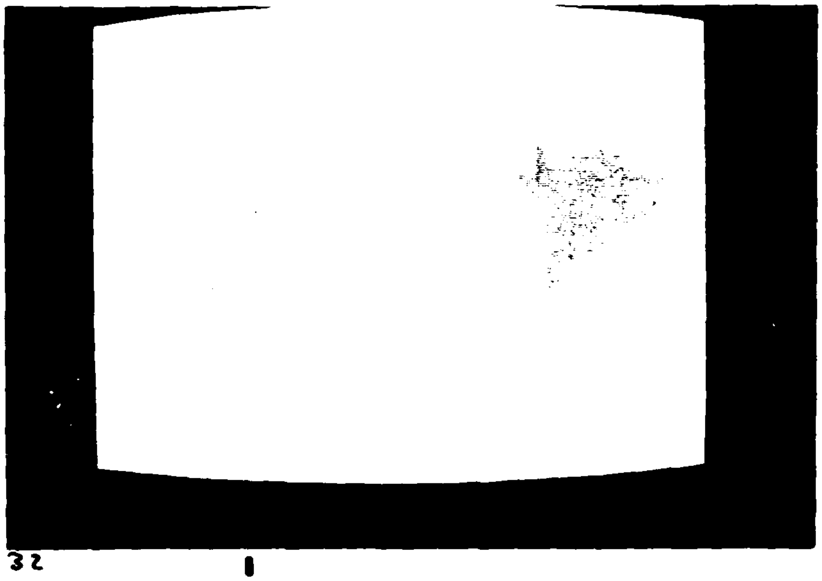


Figure 15. Filtered Masterscene: $H(f) = 1$ for $0 \leq f \leq 4$, 0 Elsewhere

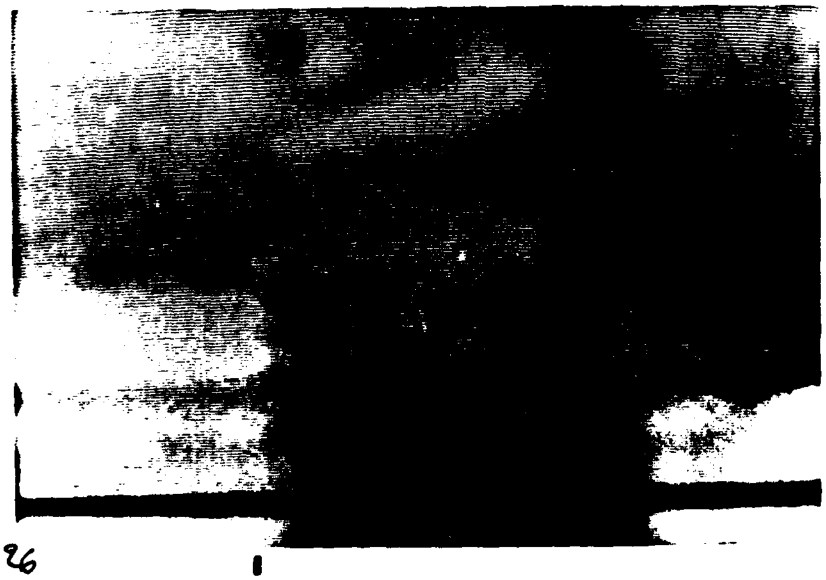


Figure 16. Filtered Masterscene: $H(f) = 1$ for $1 \leq f \leq 15$, 0 Elsewhere

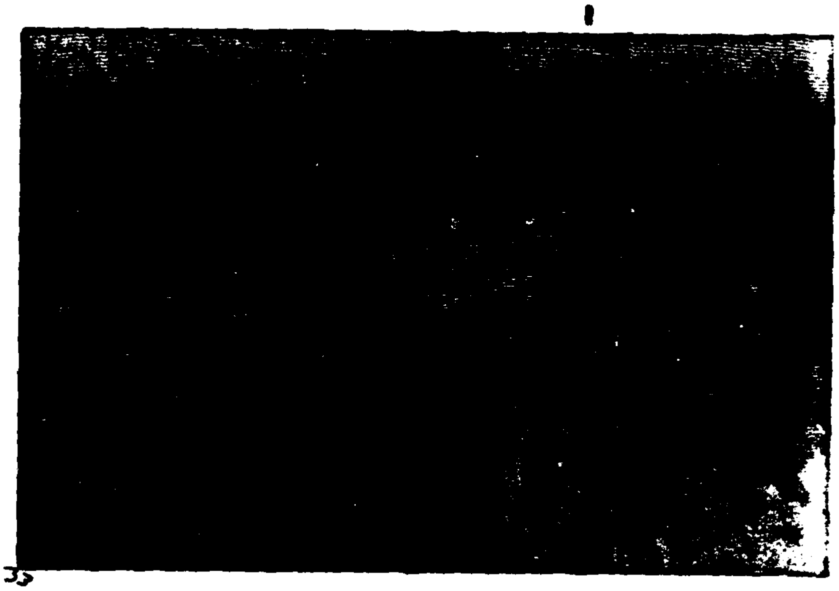


Figure 17. PIMT Image of Figure 16

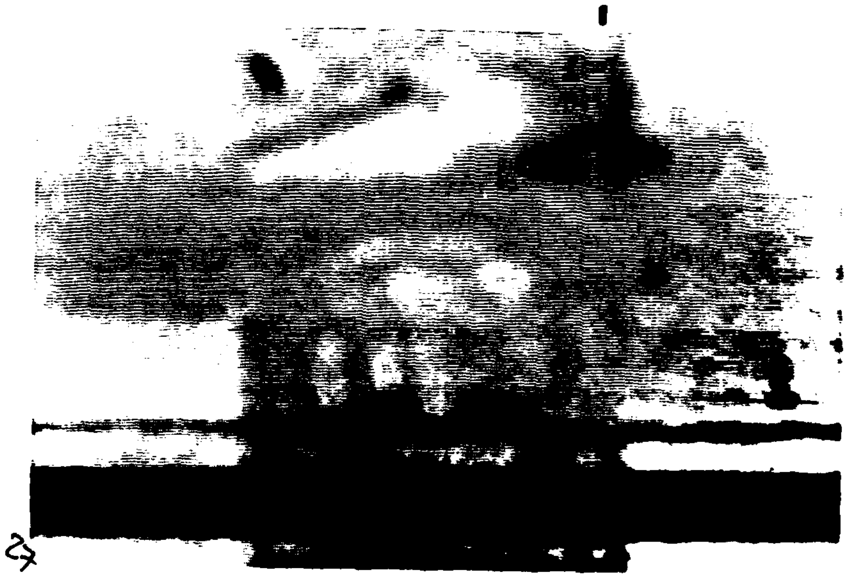


Figure 18. Filtered Masterscene: $H(f) = 1$ for $1 \leq f \leq 25$, 0 Elsewhere

34

Figure 19. PIMT Image of Figure 18

28

Figure 20. Filtered Masterscene:
$H(f) = 1$ for $10 \leq f \leq 25$, 0 Elsewhere

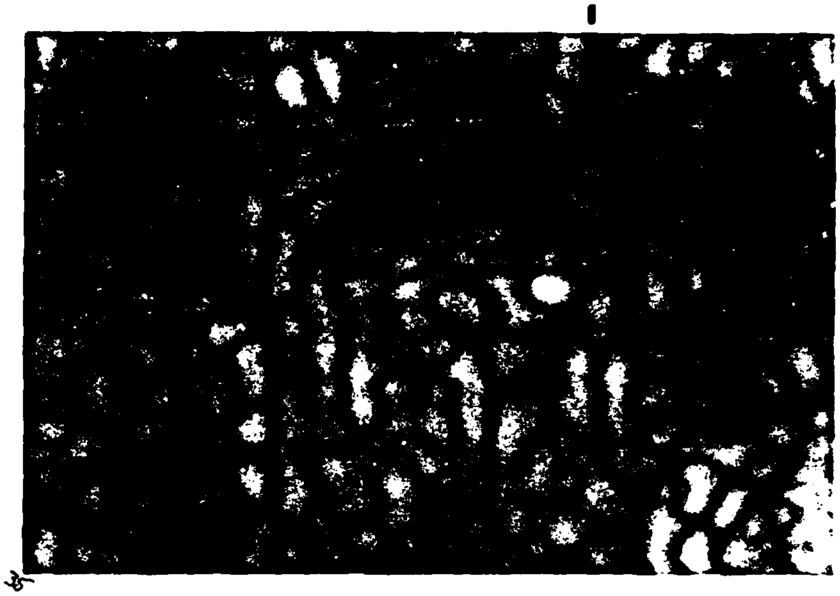


Figure 21. PIMT Image of Figure 20

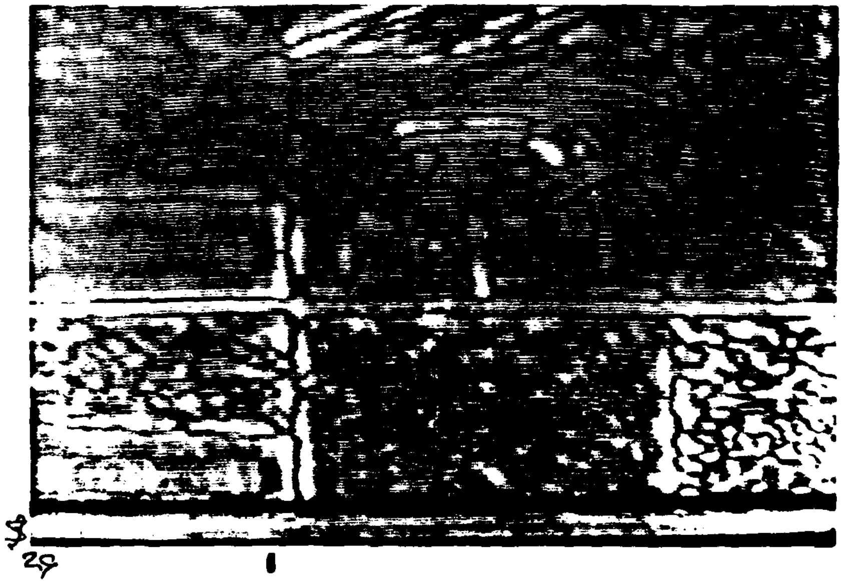


Figure 22. Filtered Masterscene:
H(f) = 1 for 10 ≤ f ≤ 50, 0 Elsewhere

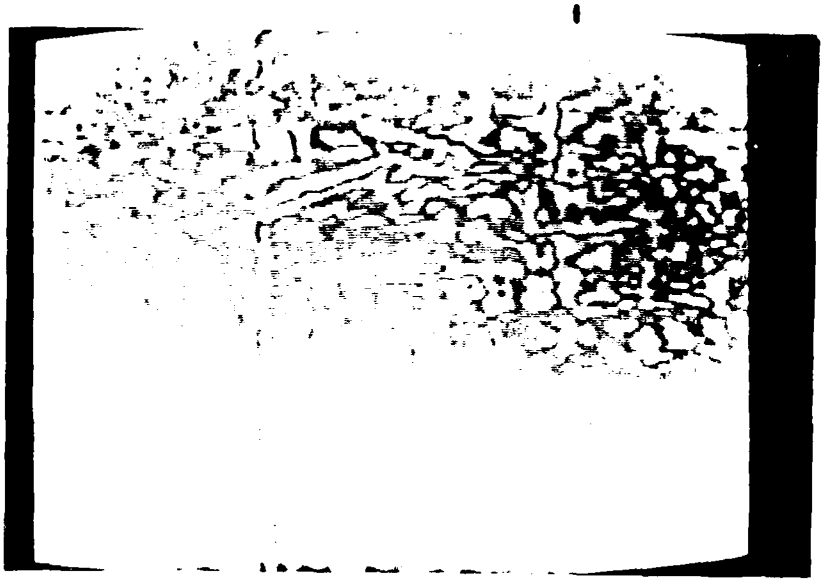


Figure 23. PIMT Image of Figure 22

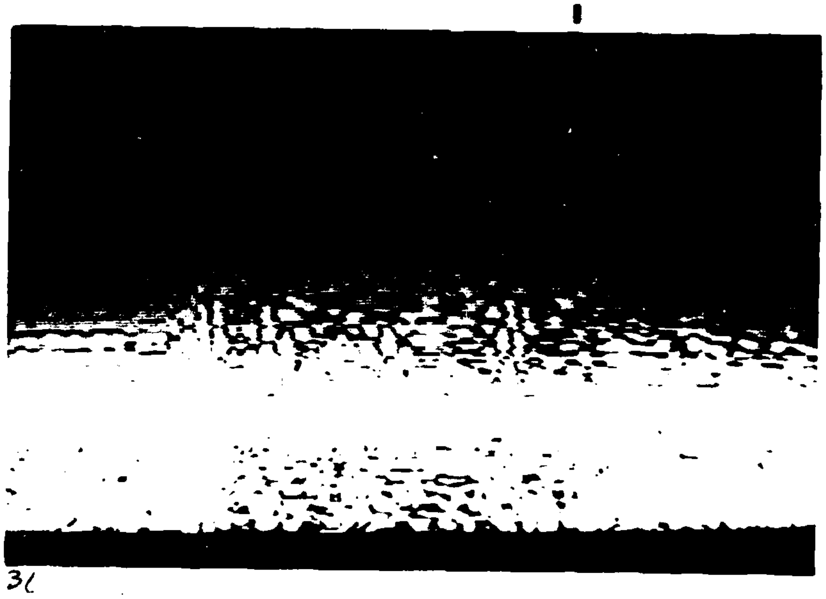
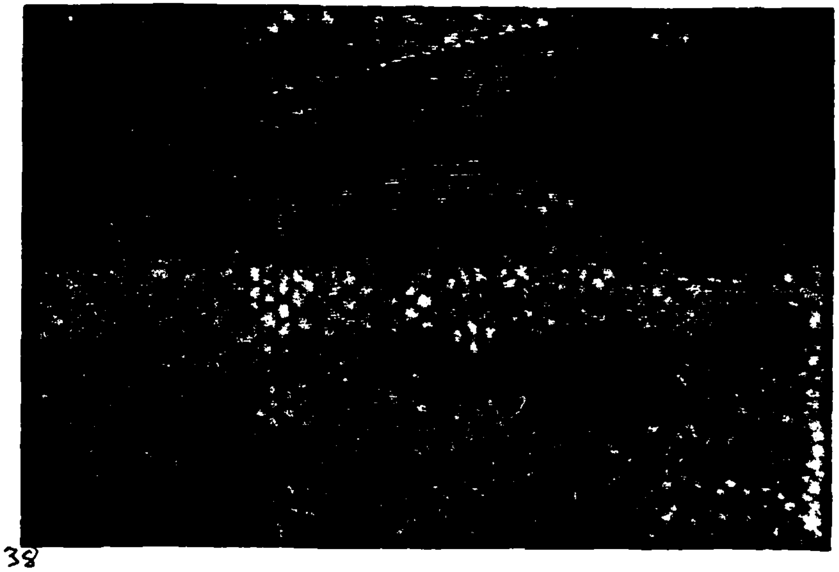
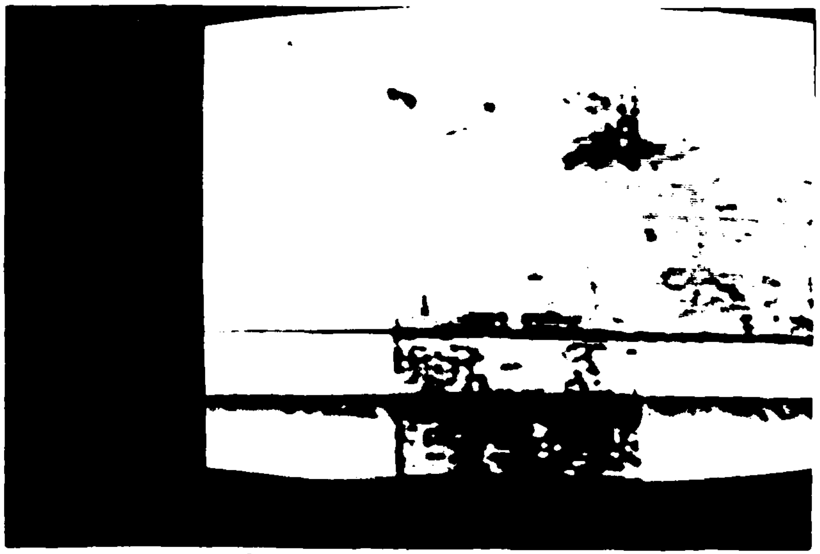


Figure 24. Filtered Masterscene:
H(f) = 1 for $25 \leq f \leq 50$, 0 Elsewhere



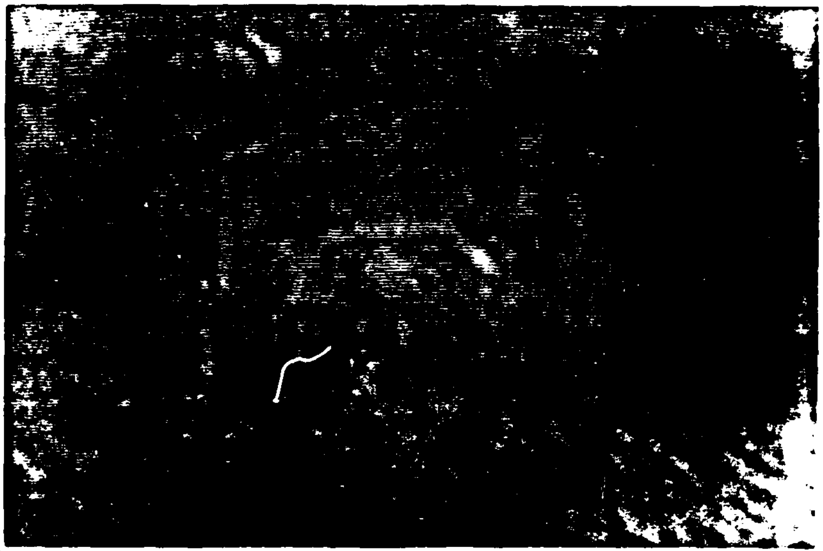
38

Figure 25. PIMT Image of Figure 24



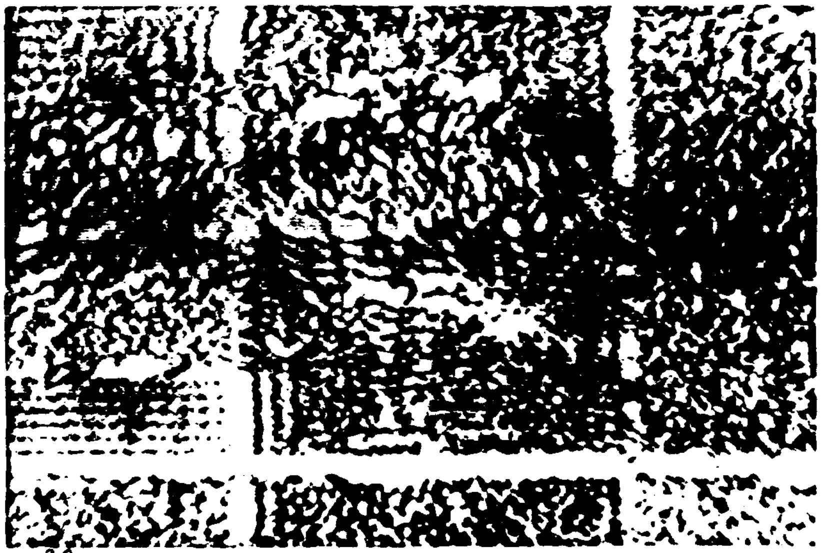
30

Figure 26. Filtered Masterscene:
$H(f) = 1$ for $1 \le f \le 50$, 0 Elsewhere



37

Figure 27. PIMT Image of Figure 26



35

Figure 28. Filtered Masterscene:
$H(f) = 1$ for $100 \leq f \leq 256$, 0 Elsewhere

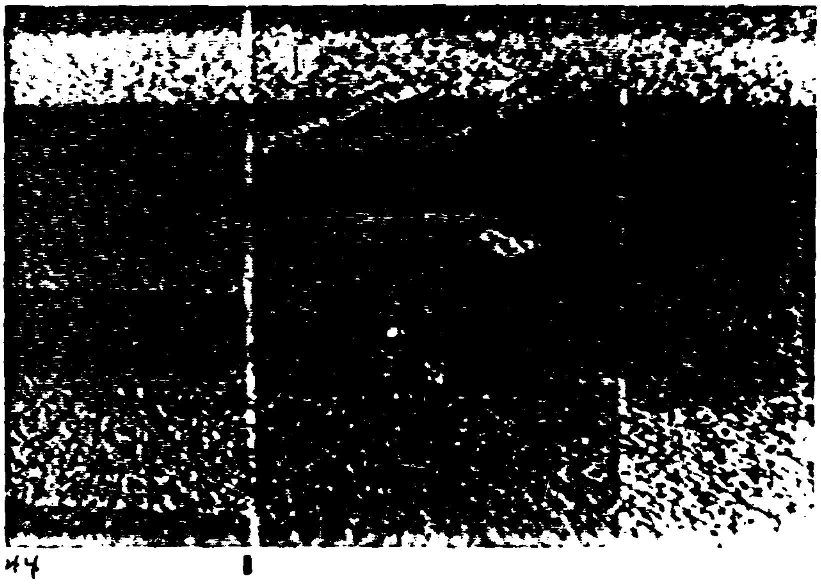


Figure 29. Inverse 2-D DFT of Masterscene with $|S(\zeta,\eta)| = 1$

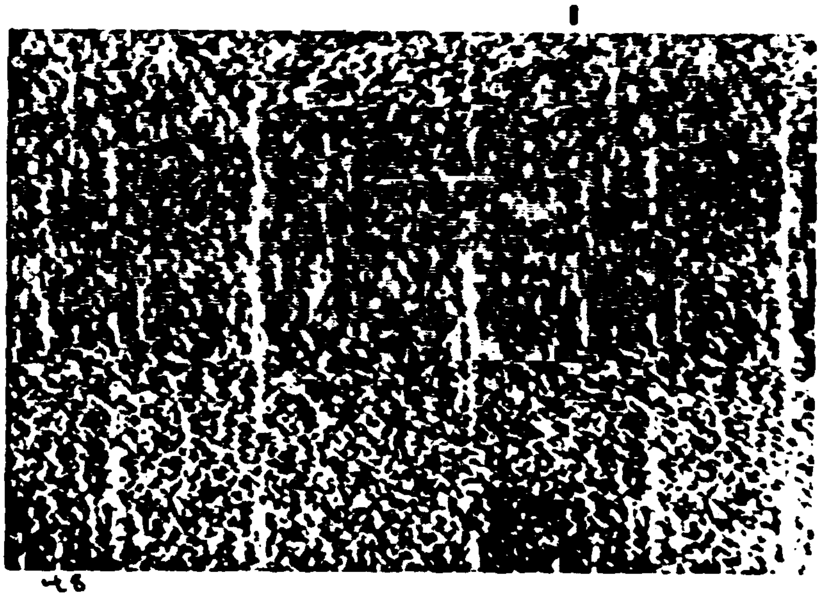
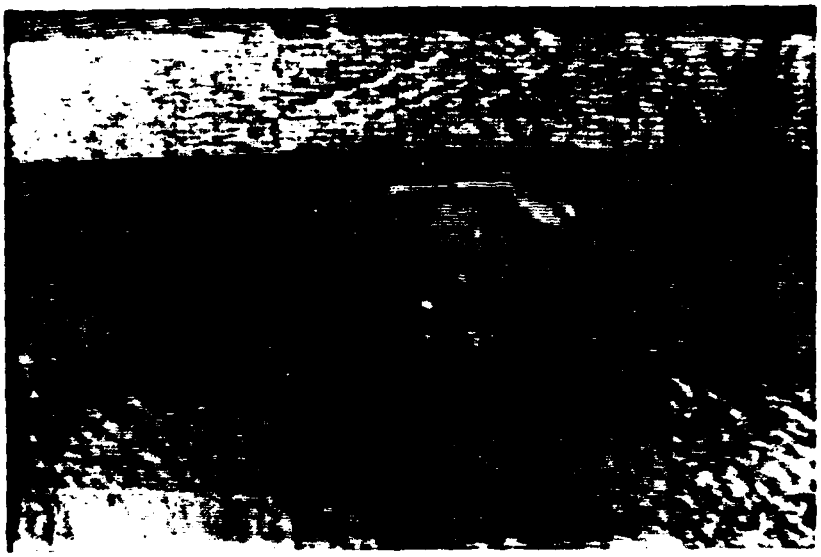
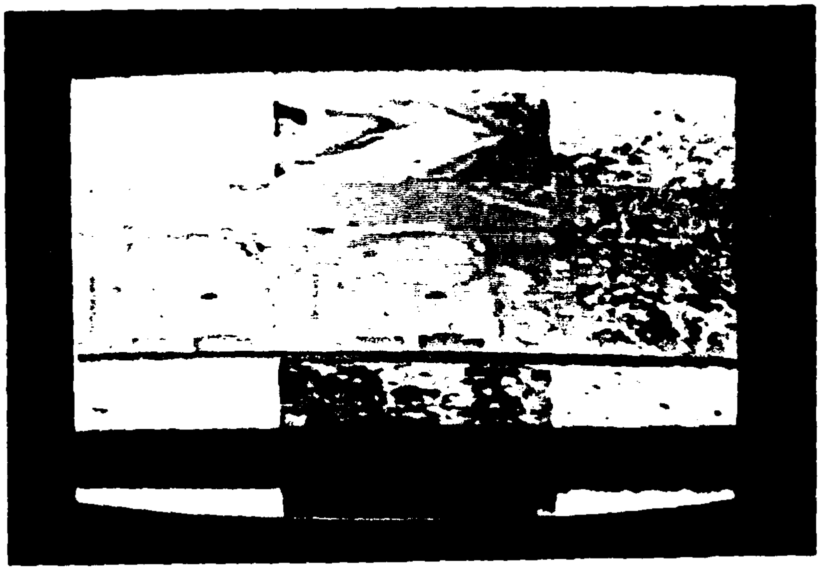


Figure 30. PIMT Image of Masterscene Using Equation 21



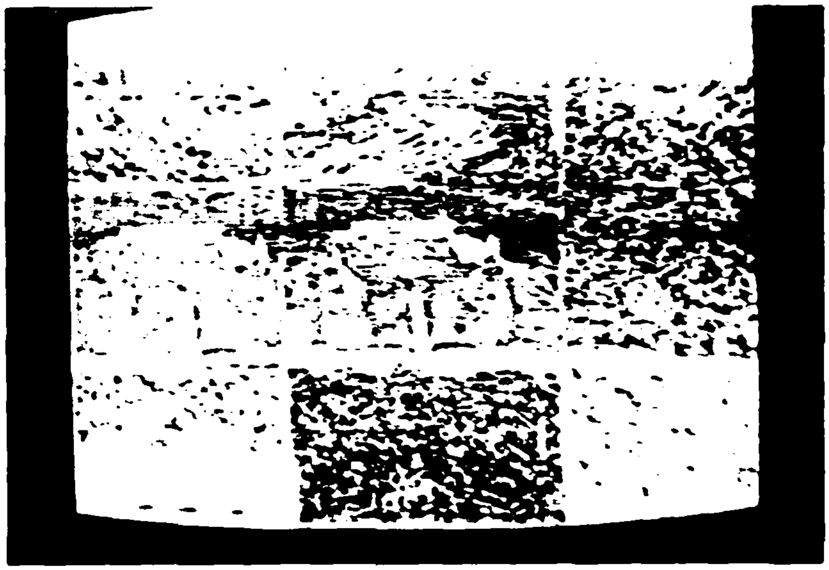
43

Figure 31. PIMT Image of Masterscene: H(f) = 1



40

Figure 32. Scene with Two Targets

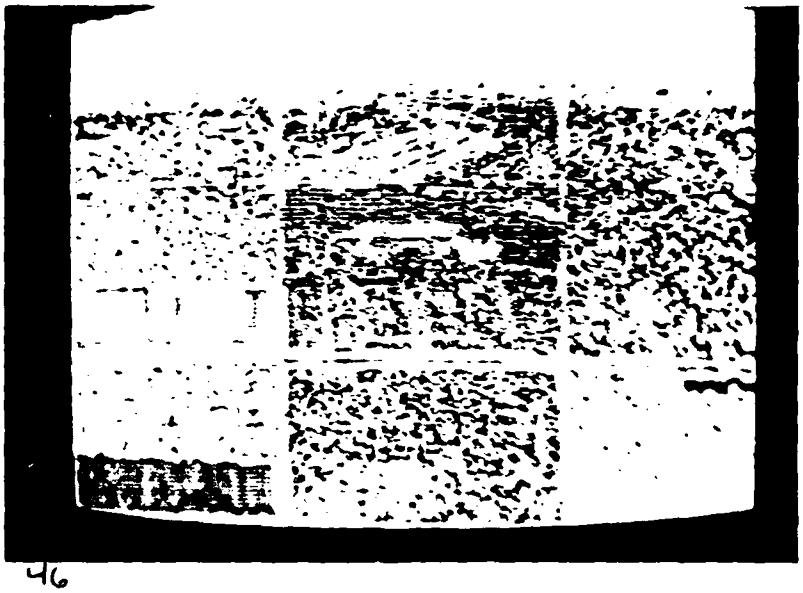


41

Figure 33. Filtered Scene of Figure 32: H(f) = f

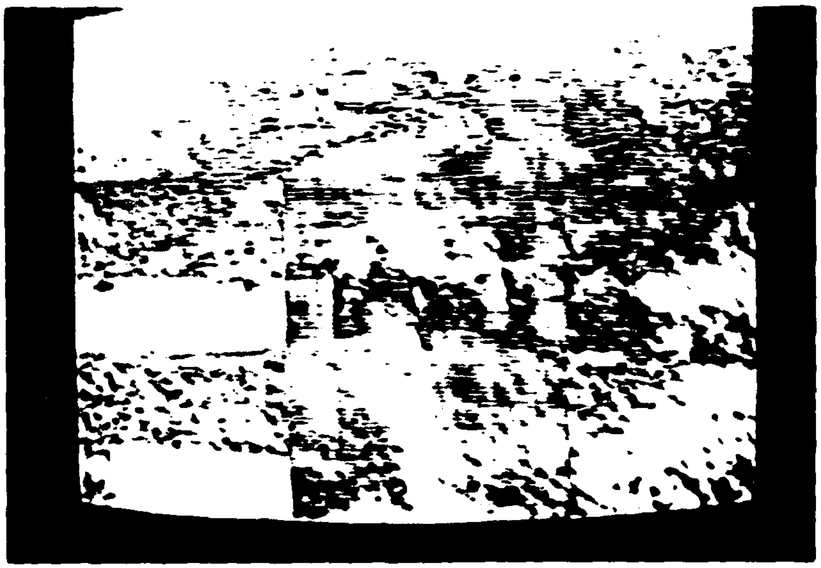
42

Figure 34. PIMT Image of Figure 33



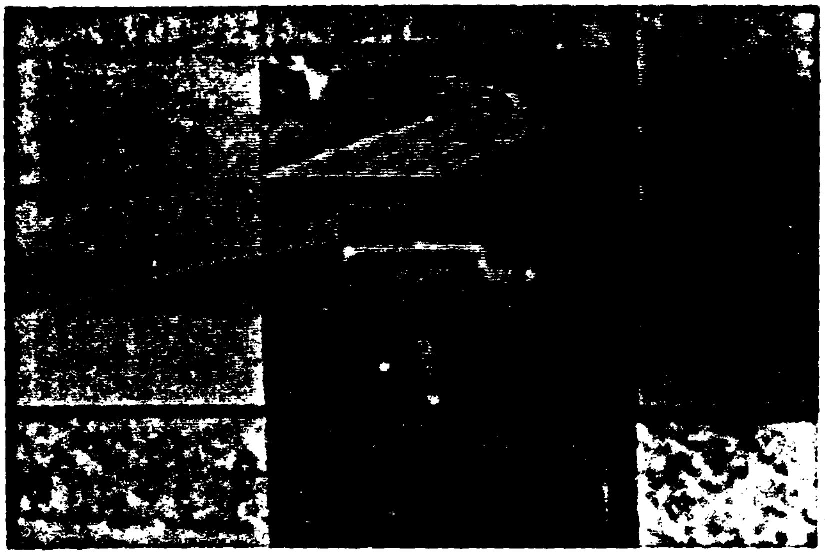
46

Figure 35. Filtered Masterscene: $H(f) = f^2$



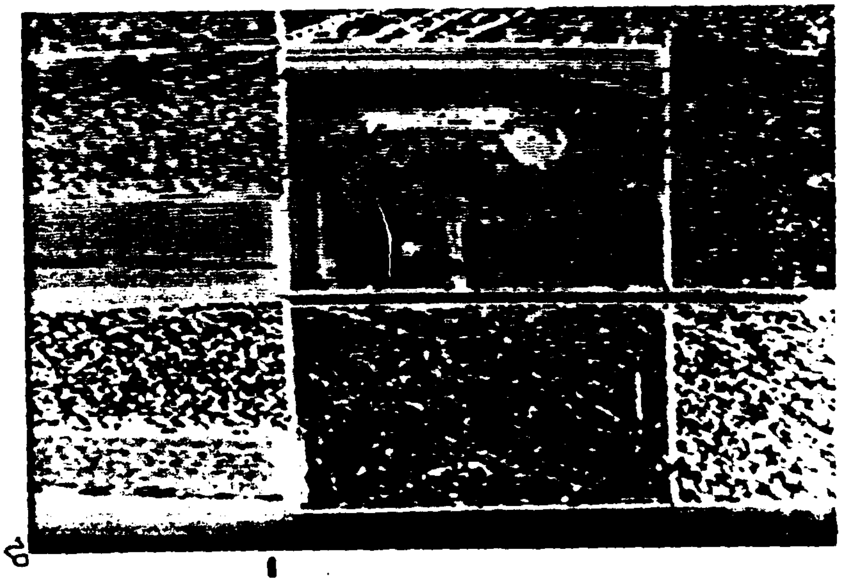
47

Figure 36. PIMT Image of Figure 35



16

Figure 37. Masterscene with Different Target (Center)

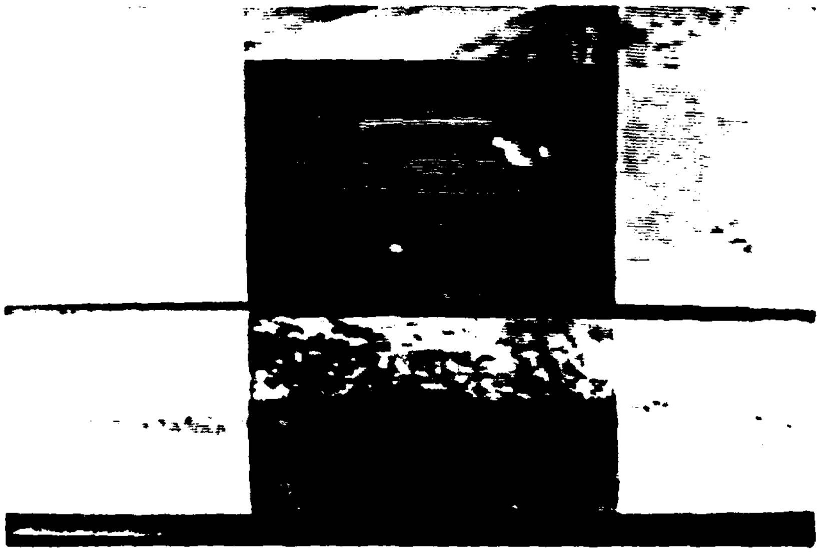


8

Figure 38. Filtered Scene of Figure 37: $H(f) = f^2$

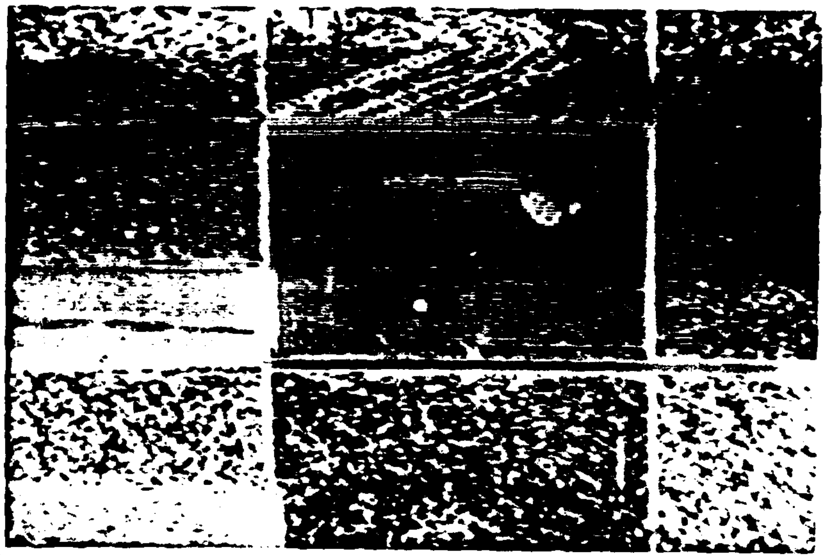
24.

Figure 39. PIMT Image of Figure 38



14

Figure 40. Scene with Different Target (1)



19

Figure 41. Filtered Scene of Figure 40: H(f) = f

23

Figure 42. PIMT Image of Figure 41

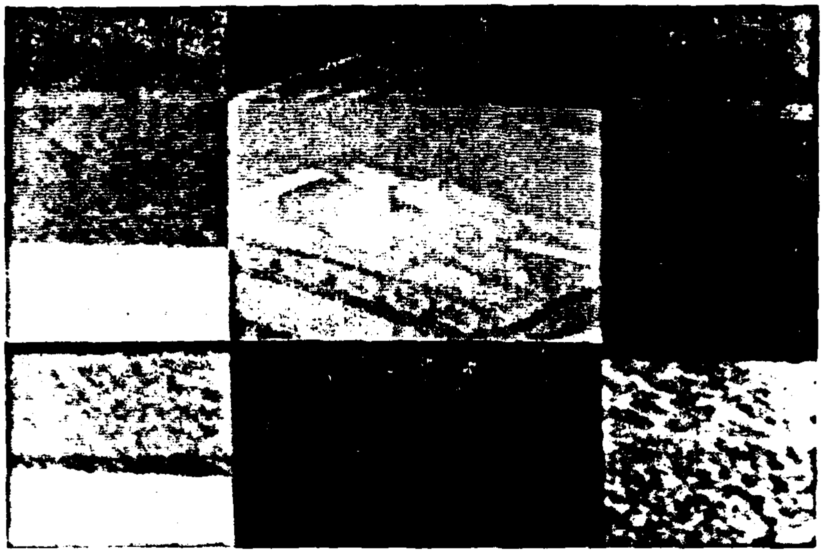


Figure 43. Scene with Different Target (2)

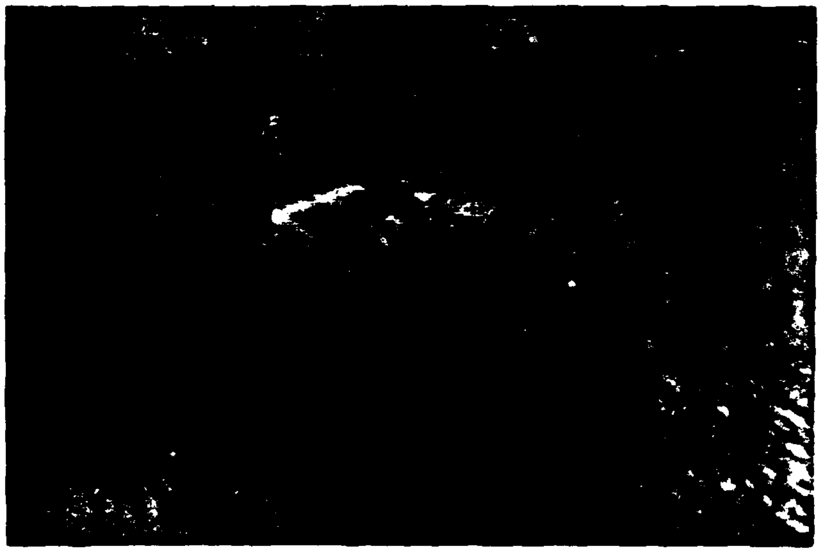
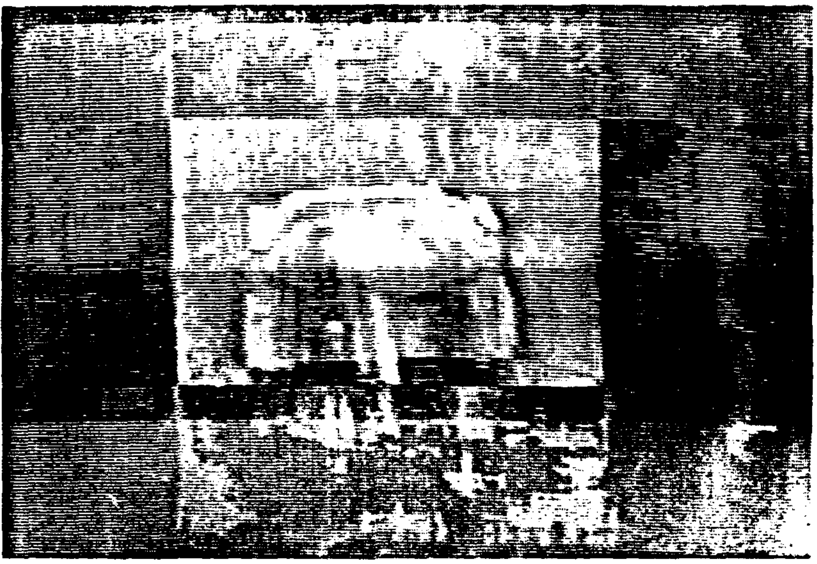
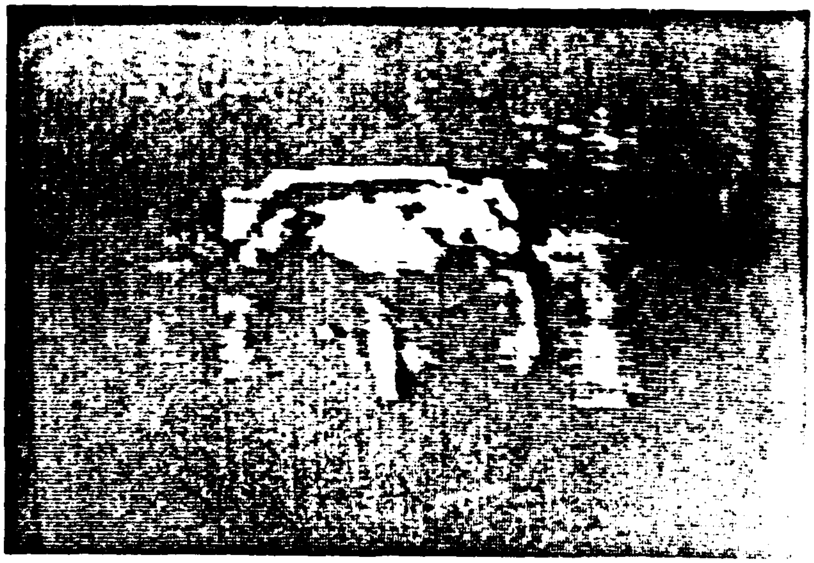


Figure 44. PIMT Image of Figure 43: H(f) = 1



2

Figure 45. PIMT Image of Figure 2: H(f) = 1/f



4

Figure 46. PIMT Image of Figure 2: H(f) = f

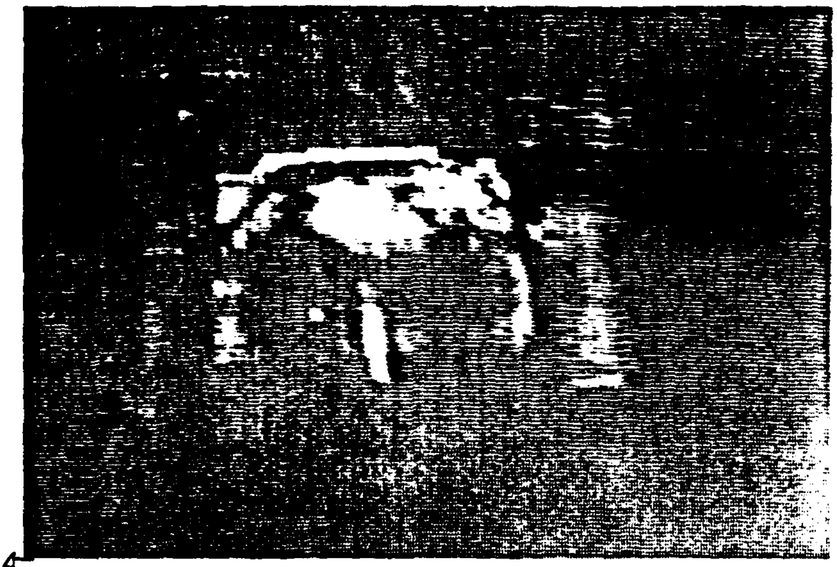


Figure 47. PIMT Image of Figure 2: $H(f) = f^2$

VII Size and Rotation Studies

The next two chapters present the results of studies made to determine the discrimination abilities of the PIMT process. These studies are concerned with how much size and rotational deviation of the target from the template being used is allowed by the PIMT process. This chapter presents the results of target size and rotation deviation studies. The next chapter presents the results on how well the PIMT process can discriminate one target from another (i.e. a truck from a tank).

Procedure

The only procedure used for these studies was the PIMT process. No filtering of the scene or template images was accomplished. The scene and template images were created from the black and white photograph shown in Figure 6. The photograph was digitized by the camera located at various distances from the photograph, resulting in images of different size. Rotated images were created by rotating the photograph in five degree increments from zero to ninety degrees, keeping the size of the photograph constant.

Figure 48 shows the template image used for the first test. Figures 49 and 50 show the scene image and the resulting PIMT image. As can be seen, most of the details of the original scene are retained in the PIMT image even though the template used was approximately 1/10 smaller.

For the next test, the template of Figure 51 and the scene of Figure 53 below were processed. The target in this scene is 1/3 larger and rotated 45 degrees from the template used. For comparison, a PIMT

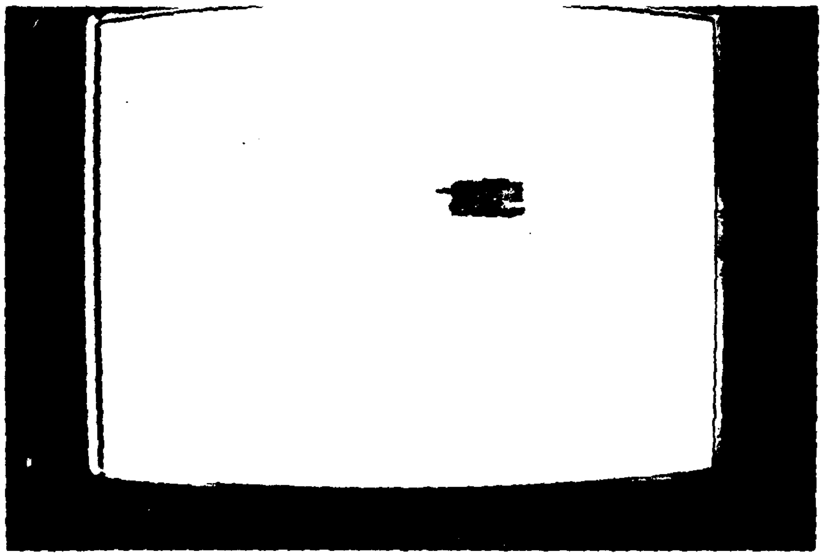


Figure 48. Template Used for Size Study

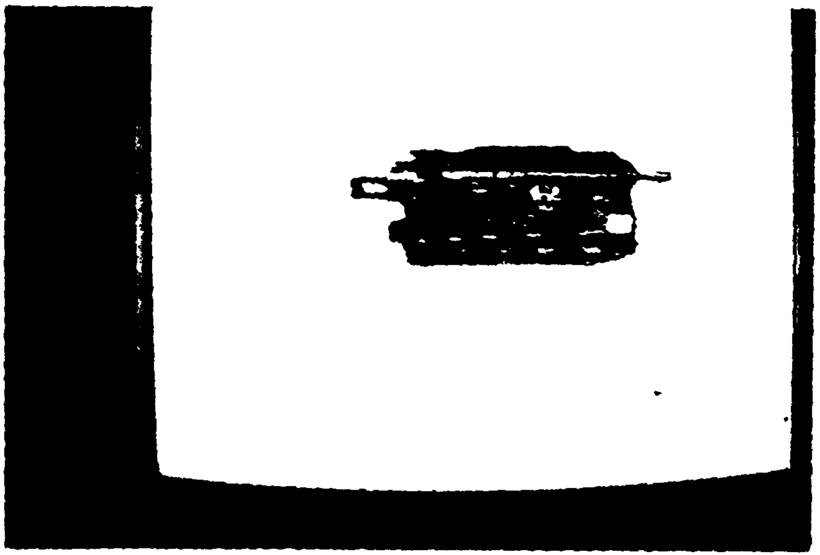


Figure 49. Scene Image Used for Size Study

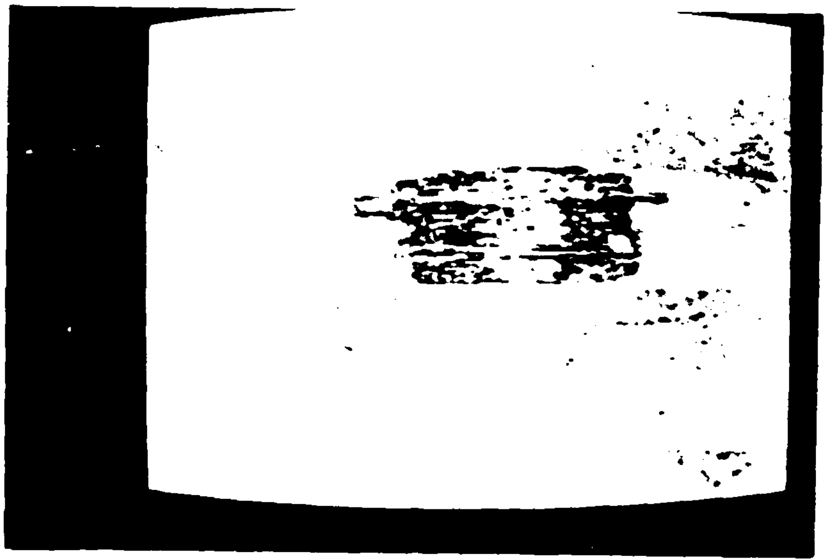


Figure 50. PIMT Image Used for Size Study

image was also created using the same scene (Figure 53) and the template of the rotated target in the scene as shown in Figure 52. The resulting PIMT images are shown in Figures 54 and 55, respectively.

Again, it can be seen in Figure 54 that the clutter is reduced and the target is visible using a target three times larger than the template used.

Discussion

This study demonstrates that although optimal PIMT clutter reduction and potential target identification occurs when the template is the same size and rotation as the target in the scene, changes in size and rotation still result in successful clutter reduction. The target is distorted, but not to the point that potential target identification would not occur. It can be concluded that severe size and rotation deviations are allowed by the PIMT process.

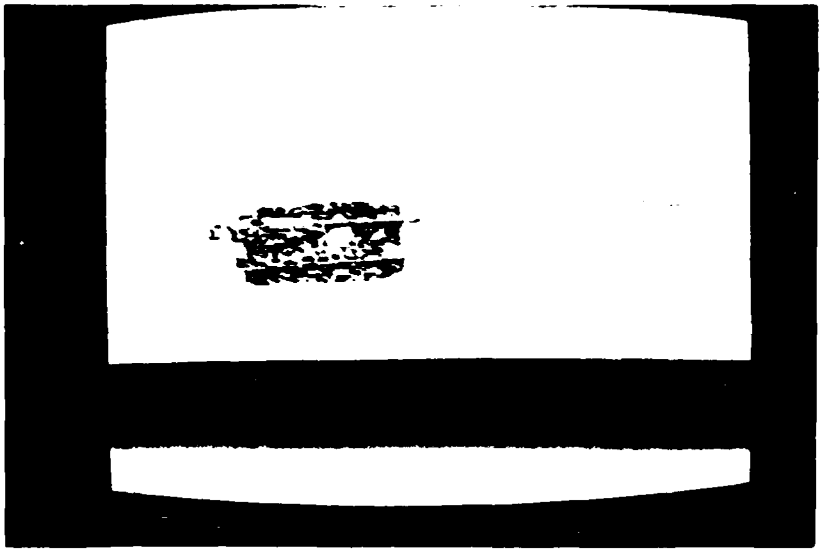


Figure 51. Template of Reduced and Unrotated Target

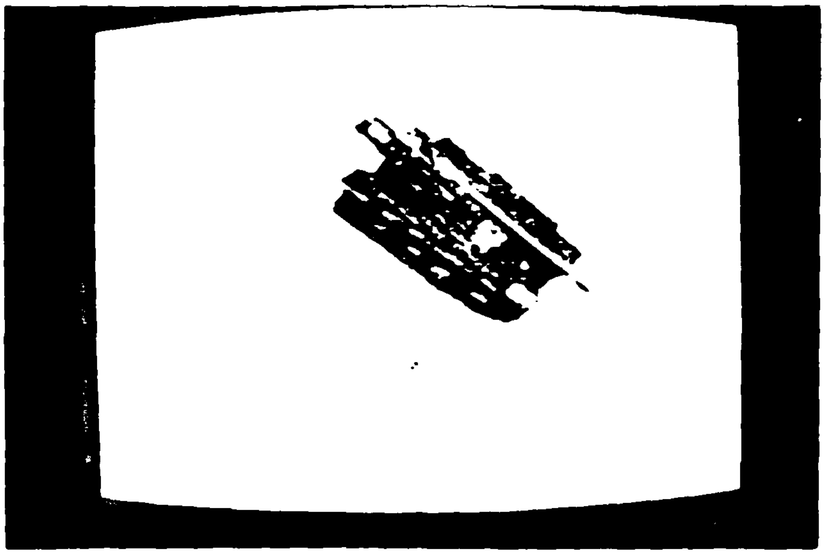


Figure 52. Template of Rotated Target

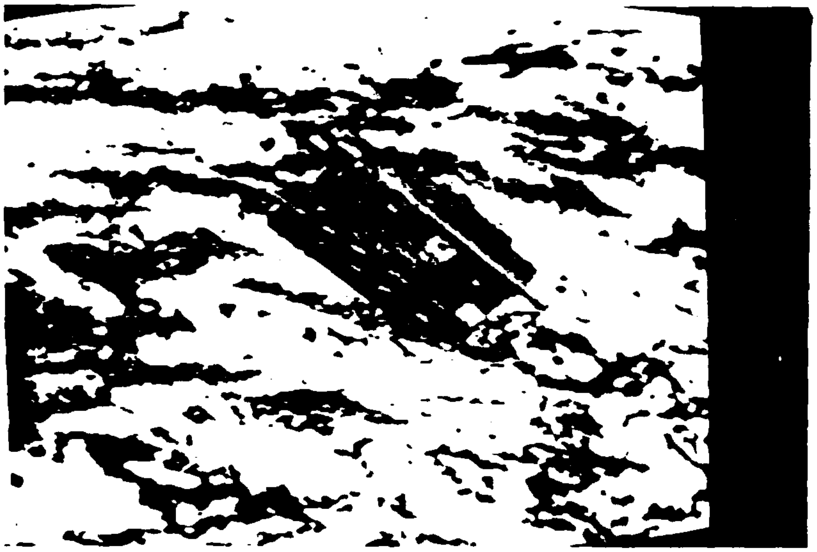


Figure 53. Scene Image with Rotated Target

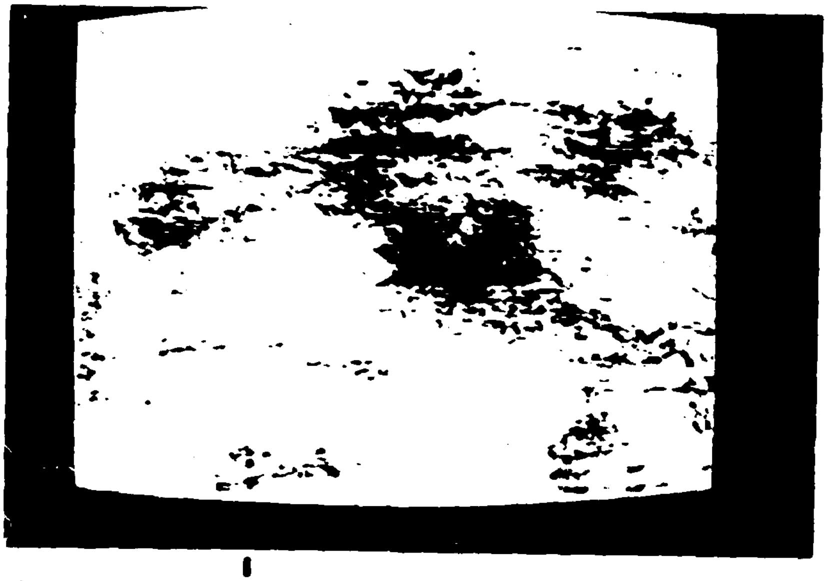


Figure 54. PIMT Image of Unrotated Template

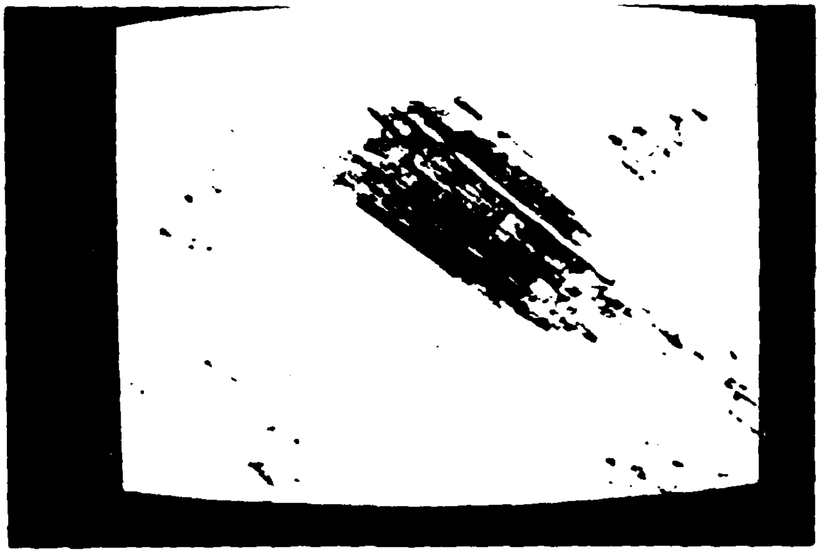


Figure 55. PIMT Image of Rotated Template

VIII Discrimination Studies

This chapter is a report on a study made to determine the discrimination ability of the PIMT process. The intent was to see if the process could differentiate between a truck and a tank or any other weapon system.

Procedure

The PIMT process was used with no filtering. The scene image used for processing is shown in Figure 56. Templates were created of each of the objects and PIMT images were obtained. Figure 57 shows the template created from the top, middle tank and figure 58 is the resulting image.

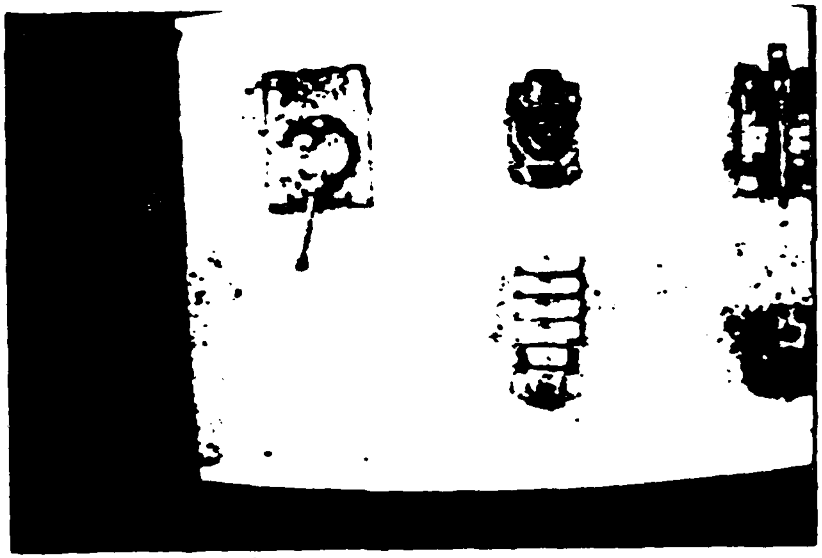


Figure 56. Scene for Discrimination Test

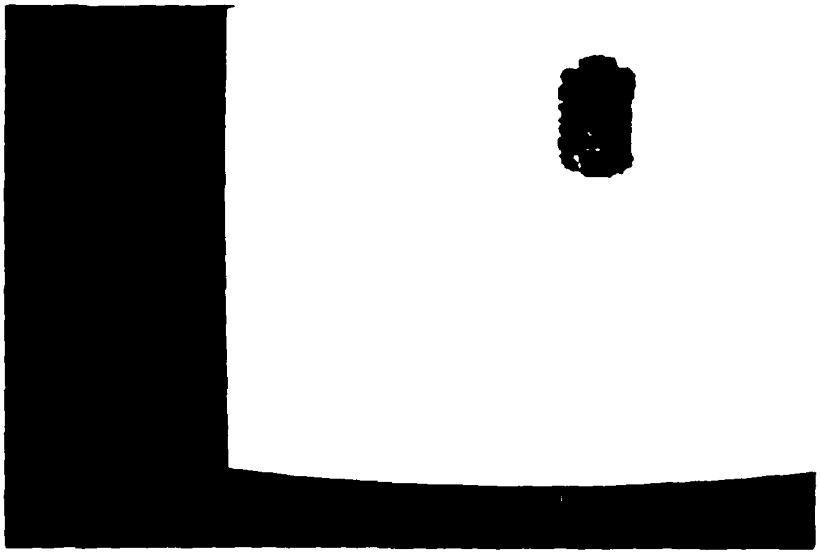


Figure 57. Template Image of Top, Middle Tank

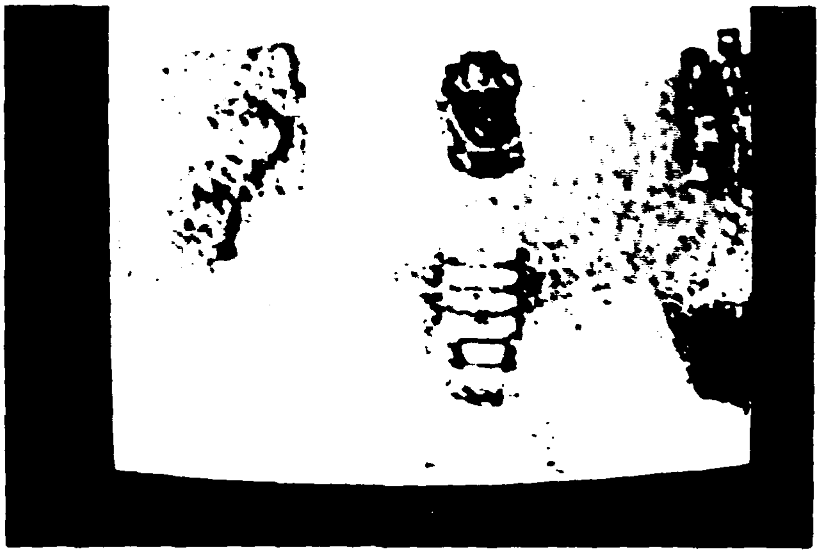


Figure 58. PIMT Image for Discrimination Test

For all templates, the PIMT images showed an equal energy distribution among the objects similar to that shown in the photograph above (i.e. the same PIMT image no matter which template was used).

It was suspected that a similar result could be obtained from a "dummy template", a template containing no details such as gun barrels or turrets (Ref 8). A template was created consisting of a rectangle of constant grey scale level that had the approximate dimensions of the objects in the scene image. The resulting PIMT image is shown in the photograph below.

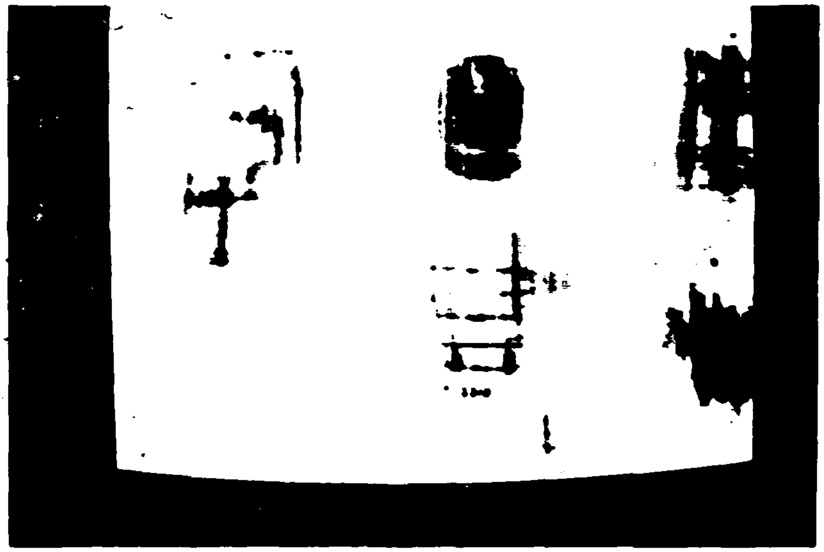


Figure 59. PIMT Image Using Dummy Template

Results

The results of this study show that the PIMT process has little or no discrimination ability.

IX Conclusion

The purpose of this thesis was to investigate the second stage of the Horev algorithm in suppressing background clutter in infrared and visible spectrum images. This thesis presents representative results obtained from processing approximately 100 infrared spectrum images and 80 visible spectrum images.

Conclusien

It can be concluded that the second stage of the Horev algorithm, the PIMT process, does suppress background clutter. This was verified by the results presented in Chapter IV.

A theory is presented in Chapter II that demonstrates the PIMT process suppresses background clutter by eliminating or reducing the frequency components contained in the scene image which are not contained in the template image. The results presented in Chapters IV through VI demonstrate that the clutter reduction is highly dependent on the scene and template images. This dependence is best demonstrated by the two "successful" images presented in chapter IV. An analysis of the template shows that most of the template energy will be concentrated in the narrow band of spatial frequencies required to produce the horizontal "lines" in the truck. A similar analysis of the scene shows that the energy will be spread over a wide range of frequencies. It can be seen in the resulting PIMT image that the spatial components retained from the original scene image are those components of the narrow band of frequencies of the template. Looking back again at the original scene, it can be seen that most of the energy in this narrow band of frequencies is concentrated at the target. This combination of favorable factors results in the successful PIMT image.

Chapter V demonstrates that when the potential target is the largest component of the scene, the PIMT process is no better than a straight correlation operation. This causes doubt as to the usefullness of the PIMT process as an initial stage of automatic pattern recognition. Chapter VI demonstrates that filtering causes no improvement of the scene image. Chapter VII demonstrates that optimal target detection and clutter reduction occurred when the template is the same size and rotation as the target in the scene image. It is also demonstrated that severe size and rotation deviations of the template to the target are allowed by the PIMT process. Finally, it is shown in Chapter VII that the PIMT process has little or no discrimination ability, making it more of a "blob" detection process than a pattern recognition process.

The results obtained in this study are consistent to those obtained by Horev (Ref 4) and Fadem and Walters (Ref 5). It is concluded that the PIMT process is unsuccessful as a single process automatic pattern recognition algorithm due to its high dependence on the scene and template images used.

Bibliography

1. Elson, B.M. "USAF Pick Lockheed Team to Develop Targeting System," Aviation Week and Space Technology, 107:23-24.

2. Gagnon, R.A. "Detection and Identification of Objects Embedded in Cluttered Fields: A Reconnaissance Problem," PhD Thesis, Air Force Inst. Tech., (1975).

3. Tou, J.T. and R.C. Gonzalez. Pattern Recognition Principles. Massachusetts: Addison-Wesley Publishing Co., 1974.

4. Horev, M. "Picture Correlation Model for Automatic Machine Recognition," M.S. Thesis, Air Force Inst. Tech., (1980).

5. Fadum, T.G. and Walters, J.J. "Target Classification Using Infrared Sensor Data," M.S. Thesis, Air Force Inst. Tech., (1980).

6. Simmons, K.A. "Machine Segmentation of Unformatted Characters," M.S. Thesis, Air Force Inst. Tech., (1981).

7. Hall, E.L. Computer Image Processing and Recognition. London: Academic Press, 1979.

8. Oppenheim, A.V. "The Importance of Phase in Signals," Proceedings of the IEEE, 69:529-541.

Appendix

Computer Programs

The following pages are the computer programs used that are unique to this thesis. This software is designed to be functional and is not optimized for speed or minimum code.

PROGRAM READTAPE

```
C          THIS PROGRAM READS DATA FOR ONE PICTURE FROM
C       THE CONVERTED TABILS FORMAT TAPE, AND STORES THE
C       DATA IN A FILE CALLED PICTURE.
C
C
C
C
        DIMENSION IO(256)
        INTEGER COMMAND,B(16)
        COMMAND=0
        CALL INIT("MTO",0,I)
         IF(I.NE.1)TYPE "INIT=",I
        CALL DFILW("PICTURE",I)
        CALL CFILW("PICTURE",3,102,I)
         IF(I.NE.1) TYPE "CFILW=",I
        CALL OPEN(2,"PICTURE",3,I,128)
         IF(I.NE.1) TYPE "OPEN=",I
        TYPE "THE TAPE CONTAINS 6 FILES WITH 40 PICTURES"
        TYPE "PER FILE.  ENTER FILE NUMBER,1,2,3,4,5,OR,6"
        ACCEPT IFILE
        TYPE "ENTER PICTURE NUMBER 1,2,3,...37,38,39,OR,40"
        ACCEPT NUM
C
C           POSITION THE TAPE TO THE DESIRED FILE
C
        GOTO(10,20,30,40,50,60),IFILE
  10    CALL MTOPD(1,"MTO:0",0,I)
        GOTO 70
  20    CALL MTOPD(1,"MTO:1",0,I)
        GOTO 70
  30    CALL MTOPD(1,"MTO:2",0,I)
        GOTO 70
  40    CALL MTOPD(1,"MTO:3",0,I)
        GOTO 70
  50    CALL MTOPD(1,"MTO:4",0,I)
        GOTO 70
  60    CALL MTOPD(1,"MTO:5",0,I)
  70    CONTINUE
        IF(NUM.EQ.1) GOTO 101
C
C           SET TAPE TO DESIRED PICTURE. CALL TO WORD TELLS
C       MTDIO TO SKIP RECORDS.  NUM*102 IS NUMBER OF
C       RECORDS SKIPPED.  SEE MTDIO COMMAND IN FORTRAN IV
C       MANUAL.
C
        CALL WORD(COMMAND,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0)
        COMMAND=COMMAND+(NUM-1)*102
        CALL MTDIO(1,COMMAND,IO,ISTAT,I,IRC)
        TYPE "RECORDS SKIPPED=",IRC
        TYPE "ISTAT=",ISTAT
        IF(I.NE.1) TYPE "MTDIO1=",I,"ISTAT=",ISTAT
```

```
 101 COMMAND=0
        DO 100 K=1,102
        KK=K-1
C
C            READ A RECORD OFF TAPE INTO ARRAY IO
C
           CALL MTDIO(1,COMMAND,IO,ISTAT,I,IRC)
            IF(I.NE.1) TYPE "MTDIO=",I,"ISTAT=",ISTAT
X            TYPE "RECORD",K,"  ",IRC,"WORDS READ"
C
C            DELETE SOME GARBAGE INSERTED BY MTDIO COMMAND
C
        DO 88 JK=207,255
  88    IO(JK)=0
C
C           INSERT CARRIAGE RETURNS EVERY 128 CHARACTER,
C       THE CALL TO WORD28 SETS THE LAST 8 BITS TO 00001101
C       (BINARY FOR CR).
C
           DO 110 J=64,256,64
        CALL WORD28(IO(J),0,0,0,0,1,1,0,1)
 110    CONTINUE
C
C        WRITE A BLOCK (ARRAY IO) ONTO FILE NAMED PICTURE
C
           CALL WRBLK(2,KK,IO,1,IW)
            IF(IW.NE.1) TYPE "WRBLK=",IW
 100       CONTINUE
        CALL CLOSE(1,IC)
         IF(IC.NE.1) TYPE "CLOSE=",IC
        CALL RLSE("MTO",IM)
         IF(IM.NE.1) TYPE "RLSE=",IM
        STOP  "ONE PICTURE READ"
        END
```

PROGRAM WORD

```
C
C          Subroutine WORD sets the bits in a 16 bit word.
C     If B15=1 bit 15 is set, if B15=0 bit 15 is cleared.
C
C
C
           SUBROUTINE WORD(I,B15,B14,B13,B12,B11,B10,B9,B8,
        +  B7,B6,B5,B4,B3,B2,B1,B0)
           INTEGER B0,B1,B2,B3,B4,B5,B6,B7,B8,B9,B10,B11,
        +  B12,B13,B14,B15
           IF(B0.EQ.0)GOTO 10
           CALL BSET(I,0)
           GOTO 11
  10       CALL BCLR(I,0)
  11       IF(B1.EQ.0) GOTO 20
           CALL BSET(I,1)
           GOTO 21
  20       CALL BCLR(I,1)
  21       IF(B2.EQ.0) GOTO 30
           CALL BSET(I,2)
           GOTO 31
  30       CALL BCLR(I,2)
  31       IF(B3.EQ.0) GOTO 40
           CALL BSET(I,3)
           GOTO 41
  40       CALL BCLR(I,3)
  41       IF(B4.EQ.0) GOTO 50
           CALL BSET(WORD,4)
           GOTO 51
  50       CALL BCLR (I,4)
  51       IF(B5.EQ.0) GOTO 60
           CALL BSET(I,5)
           GOTO 61
  60       CALL BCLR(I,5)
  61       IF(B6.EQ.0) GOTO 70
           CALL BSET(I,6)
           GOTO 71
  70       CALL BCLR(I,6)
  71       IF(B7.EQ.0) GOTO 80
           CALL BSET(I,7)
           GOTO 81
  80       CALL BCLR(I,7)
  81       IF(B8.EQ.0) GOTO 90
           CALL BSET(I,8)
           GOTO 91
  90       CALL BCLR(I,8)
  91       IF(B9.EQ.0) GOTO 100
           CALL BSET(I,9)
           GOTO 101
 100       CALL BCLR(I,9)
 101       IF(B10.EQ.0) GOTO 110
```

```
     CALL BSET(I,10)
        GOTO 111
110     CALL BCLR(I,10)
111     IF(B11.EQ.0) GOTO 120
        CALL BSET(I,11)
        GOTO 121
120     CALL BCLR(I,11)
121     IF(B12.EQ.0) GOTO 130
        CALL BSET(I,12)
        GOTO 131
130     CALL BCLR(I,12)
131     IF(B13.EQ.0) GOTO 140
        CALL BSET(I,13)
        GOTO 141
140     CALL BCLR(I,13)
141     IF(B14.EQ.0) GOTO 150
        CALL BSET(I,14)
        GOTO 151
150     CALL BCLR(I,14)
151     IF(B15.EQ.0) GOTO 160
        CALL BSET(I,15)
        GOTO 161
160     CALL BCLR(I,15)
161     CONTINUE
        RETURN
        END
```

PROGRAM WORD28

```
C
C       Subroutine WORD28 sets the last 8 bits in a 16 bit
C  word.  If B0=0 bit 0 will be cleared.  If B0=1 bit 0
C  will be set.
C
C
C
        SUBROUTINE WORD28(IWORD,B7,B6,B5,B4,B3,B2,B1,B0)
        INTEGER B0,B1,B2,B3,B4,B5,B6,B7,B8,B9,B10,B11,B12,
     +  B13,B14,B15
        IF(B0.EQ.0)GOTO 10
        CALL BSET(IWORD,0)
        GOTO 11
 10     CALL BCLR(IWORD,0)
 11     IF(B1.EQ.0) GOTO 20
        CALL BSET(IWORD,1)
        GOTO 21
 20     CALL BCLR(IWORD,1)
 21     IF(B2.EQ.0) GOTO 30
        CALL BSET(IWORD,2)
        GOTO 31
 30     CALL BCLR(IWORD,2)
 31     IF(B3.EQ.0) GOTO 40
        CALL BSET(IWORD,3)
        GOTO 41
 40     CALL BCLR(IWORD,3)
 41     IF(B4.EQ.0) GOTO 50
        CALL BSET(IWORD,4)
        GOTO 51
 50     CALL BCLR (IWORD,4)
 51     IF(B5.EQ.0) GOTO 60
        CALL BSET(IWORD,5)
        GOTO 61
 60     CALL BCLR(IWORD,5)
 61     IF(B6.EQ.0) GOTO 70
        CALL BSET(IWORD,6)
        GOTO 71
 70     CALL BCLR(IWORD,6)
 71     IF(B7.EQ.0) GOTO 80
        CALL BSET(IWORD,7)
        GOTO 81
 80     CALL BCLR(IWORD,7)
 81     RETURN
        END
```

PROGRAM READFILE

```
C     PROGRAM READFILE READS THE DATA FROM THE FILE
C    CREATED BY PROGRAM READTAPE.  THE DATA CAN NOW BE PRO-
C    CESSED AS THE USER WISHES.  THE PICTURE DATA (PIXEL
C    VALUES) IS STORED IN THE ARRAY MATRIX.  MATRIX(1,1) IS
C    THE LEFT REAR OF THE IR SCENE, MATRIX(100,1) IS THE
C    RIGHT REAR, MATRIX (100,1) IS THE LEFT FRONT, AND,
C    MATRIX(100,100) IS THE RIGHT FRONT
C
C
        COMMON MATRIX(10240)
        DIMENSION BKGND(10),AGENCY(10),DATE(3),PICCOM(35),
     +   DESC(20),INST(10),SCENE(35),COND(5),HIST(20),
        INTEGER TCODE,CMCODE,SITE,BEARING,BNCODE,BKGND,
     +   DATE,WEATHER,DESC,SCENE,COND,HIST,PICCOM,AGENCY
        INTEGER DUMMY(64),DUM(7),OUTFILE(7)
X       CALL IOF(1,MAIN,OUTFILE,I1,I2,I3,MS,I4,I5,I6,I7)
X       CALL DFILW("OUT",I)
        CALL DFILW("MATRIX",I)
        CALL OPEN(2,"PICTURE",3,128,I)
X       CALL CFILW("OUT",3,104,I)
X       CALL OPEN(3,"OUT",3,I,128)
        CALL CFILW("MATRIX",3,40,I)
        CALL OPEN(4,"MATRIX",3,I)
        READ(2,100) TCODE,BKGND,CMCODE,BNCODE,AGENCY,SITE,
     +   DATE,WEATHER,DESC,INST,IVUANG,IDEPANG,IRANGE,SCENE,
     +     IRPM,COND,(HIST(I),I=1,10),(HIST(I),I=11,20),
     +     BEARING,IWIND,IVTRAN1,IVTRAN2,IRTRAN1,IRTRAN2,
     +    IRCODE1,IRCODE2,IPYRA1,IPYRA2,IPYRHE1,IPYRHE2
 100    FORMAT(I3,10A2,2I3,10A2,I3,3A2,A1,20A2,10A2,I6/2I6,
     +    35A2,I6,5A2,10A2/10A2,2I6,10I3)
        READ(2,120) TIME,CFSLOPE,CFINTCPT,PICCOM,HORRES,
     +  VERRES,HUMID,WIND,AIRTEMP,BARPRES,RAIN,SNOW,SOIL,
     +  VISABLE
 120    FORMAT(/F10.3,2F20.15,35A2,F6.3/F6.3,2F10.2,F10.1,
     +  F10.0,4F10.2)
        READ(2,160) (MATRIX(J),J=1,10000)
 160    FORMAT(//99(3(31I4/),7I4/),3(31I4/),7I4)
X       WRITE(4,200) MATRIX
X  200  FORMAT(100(/" ",3(25I4/" ")25I4))
        CALL WRBLK(4,0,MATRIX,40,IE)
        IF(IE.NE.1) TYPE "READFILE WRBLK4=",IE
C
C          THE PROGRAM WILL SKIP OVER THE SECTION THAT
C      WRITES DATA TO THE FILE CALLED OUT.
C
        IIII=1
        IF(IIII.EQ.0)GOTO 666
        GOTO 333
C
  666   CONTINUE
```

```
        WRITE(3,110) PICCOM,TCODE,SCENE,CMCODE
  110   FORMAT("0",35A2,10X,"TARGET CODE=",I3/"0",35A2,10X,
     +   "COUNTERMEASURES CODE=",I3)
        WRITE(3,111) BKGND,BNCODE,DESC,AGENCY
  111   FORMAT("0","BACKGROUND - ",10A2,47X,"BANDCODE= ",I3
     +  /"0",20A2,40X,"AGENCY-",10A2)
        WRITE(3,112) DATE,TIME,SITE,IVUANG,INST
  112   FORMAT("0","DATE-",1X,3A2,10X,"TIME-",1X,F10.3,
     +  42X,"SITE-",I3,/"0","VIEWING ANGLE-",I4," DEG",
     +  58X,"MEASURING INST ",10A2)
        WRITE(3,113) IDEPANG,IPYRA1,IPYRA2,BEARING,IPYRHE1,
     +  IPYRHE2
  113   FORMAT("0","DEPRESSION ANGLE-",I3," DEG",56X,
     +  "PYRANOMETER READING-",1X,2I3," W/M**2"/"0",
     +  "BEARING-",I4," DEG",64X,"PYRHELIOMETER READING-",
     +  1X,2I3," W/M**2")
        WRITE(3,114) IRANGE,CFSLOPE,COND,CFINTCPT
  114   FORMAT("0","RANGE-",I3," METERS",64X,
     +   "CONVERSION FACTOR-SLOPE=",F20.15/
     +   "0","PRIOR VEHICLE STATE-",5A2,50X,"CONVERSION
     +  FACTOR-INTERCEPT=", F20.15)
        WRITE(3,115) HIST,SOIL,IWIND,WIND
  115   FORMAT("0",20A2,40X,"SOIL MOISTURE CONTENT%",F10.2/
     +  "0","WIND DIRECTION-",I4," DEG",7X,"WINDSPEED",
     +  F10.2," KNOTS")
        WRITE(3,116) IVTRAN1,IVTRAN2,IRTRAN1,IRTRAN2,
     +  VISABLE
  116   FORMAT("0","VISABLE TRANSMISSION-",2I3," %"/"0",
     +  "IR TRANSMISSION-",2I3,1X,"%"/"0","VISABILITY",
     +  F10.2,1X,"KM")
        WRITE(3,117) HUMID,AIRTEMP,BARPRES,RAIN,SNOW
  117   FORMAT("0","RELATIVE HUMIDITY-",F10.2,1X,"%"/"0",
     +  "AIR TEMPERATURE-",1X,F10.1," DEG CENTIGRADE"/"0",
       "BAROMETRIC PRESSURE",F10.0," MILLIBARS"/"0",
     +  "RAIN RATE-",F10.2," MM PER HOUR"/"0",
     +  "SNOW TEMPERATURE-",1X,F10.2,"DEG CENTIGRADE")
        WRITE(3,118) HORRES,VERRES
  118   FORMAT("0","HORIZONTAL RESOLUTION",
     +   1X,F6.3," MRAD",10X,"VERTICLE RESOLUTION ",F6.3,
     +  " MRAD")
        WRITE(3,122)
  122   FORMAT("0",45X,"A VALUE OF 999999 INDICATES NO",
     +  " DATA AVAILABLE")
X       DO 121 K=1,100
X       WRITE(3,119) K,(MATRIX(J,K),J=1,100)
X  121  CONTINUE
X  119  FORMAT("0","ROW",I4,/" ",3(25I5/" "),25I5)
  333   CONTINUE
X       WRITE(10,99)OUTFILE(1)
X  99   FORMAT(" ",S13,"created by READFILE")
        TYPE"MATRIX CREATED BY READFILE"
        CALL RESET
        END
```

PROGRAM RMAT

```
C
C          Program RMAT converts a 100x100 TABILS format
C       picture  matrix into a 100x100 real matrix
C
C
C
C
        DIMENSION IO(512),MATRIX(10240)
        REAL MA(256)
        COMMON MA
        EQUIVALENCE(MA(1),IO(1))
        INTEGER OUTFILE(7),INFILE(7)
        CALL IOF(2,MAIN,INFILE,OUTFILE,I1,I2,MS,I3,I4,I5,I6)
C
  2     FORMAT(" ",S14," created by RMAT")
C
        CALL OPEN(4,INFILE,1,IE)
        IF(IE.NE.1) TYPE "RMAT OPEN4=",IE
        CALL DFILW(OUTFILE,IE)
        IF(IE.NE.1) TYPE "RMAT DFILW=",IE
        CALL CFILW(OUTFILE,3,80,IE)
        IF(IE.NE.1) TYPE "RMAT CFILW=",IE
        CALL OPEN(5,OUTFILE,3,IE)
        IF(IE.NE.1) TYPE "RMAT OPEN5=",IE
C
X 910   FORMAT(100(/3(25I4/)25I4))
X       READ(4,910) MATRIX
        CALL RDBLK(4,0,MATRIX,40,IE)
        IF(IE.NE.1) TYPE "RMAT RDBLK4=",IE
C
        SUM=0.0
        DO 11 J=1,10000
        SUM=SUM+MATRIX(J)
  11    CONTINUE
        SUM=SQRT(SUM)
        KK=0
        DO 100 I=1,40
        DO 200 J=1,256
 200    RA(J)=0.0
        DO 400 J=1,256
        KK=KK+1
 400    MA(J)=MATRIX(KK)/SUM
 300    CALL WRBLK(5,((I*2)-2),IO,2,IE)
        IF(IE.NE.1)TYPE "RMAT WRBLK=",IE
 100    CONTINUE
        WRITE(10,2) OUTFILE(1)
        CALL RESET
        END
```

PROGRAM PIX1

```
C
C            Program PIX1 converts an input 256x256 real
C    number file into a VIDEO file.
C
C       Version 2
C
C
        INTEGER OUTFILE(7)
        DIMENSION IO(2048),IO1(1024),IO2(256),INFILE(7)
        COMMON RMAG(1024)
        EQUIVALENCE(RMAG(1),IO(1))
C
        CALL IOF(2,MAIN,INFILE,OUTFILE,I1,I2,MS,I3,I4,I5,I6)
C
   9    FORMAT(" ",S13," created.")
C
        CALL OPEN(3,INFILE,1,IE)
        IF(IE.NE.1) TYPE"PIX1 OPEN3=",IE
        CALL DFILW(OUTFILE,IE)
        IF(IE.NE.1) TYPE"PIX1 DFILW=",IE
        CALL CFILW(OUTFILE,3,64,IE)
        IF(IE.NE.1)TYPE"PIX1 CFILW=",IE
        CALL OPEN(4,OUTFILE,3,IE)
        IF(IE.NE.1) TYPE "PIX1 OPEN4=",IE
C
C       Find min and max values of infile.
C
        RMIN=1.0E60
        RMAX=0.0
        DO 2 I=0,63
        CALL RDBLK(3,(8*I),IO,8,IE)
        IF(IE.NE.1) TYPE"PIX1 RDBLK1=",IE
        DO 3 J=1,1024
        IF(RMAG(J).GT.RMAX) RMAX=RMAG(J)
        IF(RMAG(J).LT.RMIN) RMIN=RMAG(J)
   3    CONTINUE
   2    CONTINUE
        TYPE "MIN=",RMIN,"   MAX=",RMAX
        REWIND 3
C
C       Convert reals to gray scale integers and pack.
C
        ACCEPT "PIX1 gray scale max?  ",RMAX1
X       RMAX1=350
        IF(RMAX.GT.RMAX1) RMAX=RMAX1
        ACCEPT "PIX1 gray scale min?  ",RMIN1
X       RMIN1=90
        IF(RMIN.LT.RMIN1) RMIN=RMIN1
        DO 5 I=0,63
        CALL RDBLK(3,(I*8),IO,8,IE)
        IF(IE.NE.1) TYPE "PIX1 RDBLK2=",IE
   DO 4 J=1,1024
      IF(RMAG(J).GT.RMAX) RMAG(J)=RMAX
      IF(RMAG(J).LT.RMIN) RMAG(J)=RMIN
      A=15.0*(RMAG(J)-RMIN)/(RMAX-RMIN)
      IO1(J)=IFIX(A)
      IF(IO1(1).GT.15) GOTO 900
  4   CONTINUE
      KK=0
      DO 6 K=1,256
      IO2(K)=0
      DO 7 J=1,4
      KK=KK+1
      IO2(K)=ISHFT(IO2(K),4)
  7   IO2(K)=IO2(K)+IO1(KK)
  6   CONTINUE
      CALL WRBLK(4,I,IO2,1,IE)
      IF(IE.NE.1) TYPE"PIX1 WRBLK=",IE
  5   CONTINUE
      WRITE(10,9) OUTFILE(1)
      CALL RESET
      GOTO 901
900   TYPE "PIX1 gray scale error"
 901  CONTINUE
      END
```

PROGRAM CMAT

```
C
C            Program CMAT converts a 100x100 TABILS format
C       picture  matrix into a 256x256 complex
C       array
C
C
C
C
       DIMENSION IO(1024),MATRIX(10240)
       COMPLEX MA(256)
       COMMON MA
       EQUIVALENCE(MA(1),IO(1))
       INTEGER OUTFILE(7),INFILE(7)
       CALL IOF(2,MAIN,INFILE,OUTFILE,I1,I2,MS,I3,I4,I5,I6)
C
   2   FORMAT(" ",S14," created by CMAT")
C
       CALL OPEN(4,INFILE,1,IE)
       IF(IE.NE.1) TYPE "CMAT OPEN4=",IE
       CALL DFILW(OUTFILE,IE)
       IF(IE.NE.1) TYPE "CMAT DFILW=",IE
       CALL CFILW(OUTFILE,3,1024,IE)
       IF(IE.NE.1) TYPE "CMAT CFILW=",IE
       CALL OPEN(5,OUTFILE,3,IE)
       IF(IE.NE.1) TYPE "CMAT OPEN5=",IE
C
X  910 FORMAT(100(/3(25I4/)25I4))
X      READ(4,910) MATRIX
       CALL RDBLK(4,0,MATRIX,40,IE)
       IF(IE.NE.1) TYPE "CMAT RDBLK4=",IE
C
       KK=0
       DO 100 I=1,256
       DO 200 J=1,256
  200  MA(J)=(0.0,0.0)
       IF(I.LE.78.OR.I.GT.178) GOTO 300
       DO 400 K=79,178
       KK=KK+1
       IF(MATRIX(KK).GT.1023) MATRIX(KK)=0
       A=MATRIX(KK)
  400  MA(K)=CMPLX(A,0.0)
  300  CALL WRBLK(5,((I*4)-4),IO,4,IE)
       IF(IE.NE.1)TYPE "CMAT WRBLK=",IE
  100  CONTINUE
       WRITE(10,2) OUTFILE(1)
       CALL RESET
       END
```

PROGRAM CREAD

```
C
C          PROGRAM CREAD READS VALUES FROM CMATRIX, AND
C     CONVERTS THEM INTO A MAGNITUDE AND PHASE FORMAT
C     MAGNITUDES ARE WRITTEN TO A FILE RMAG.
C     PHASE ANGLES ARE WRITTEN TO A FILE CALLED ANG.
C
C
C
C
      DIMENSION IO(1024),RMAG(256),ANG(256),IO1(512),
     + IO2(512)
      COMPLEX MA(256)
      COMMON MA,RMAG,ANG

      EQUIVALENCE(MA(1),IO(1))
      EQUIVALENCE(IO1(1),RMAG(1))
      EQUIVALENCE(IO2(1),ANG(1))
      DIMENSION INFILE(7)
      CALL IOF(1,MAIN,INFILE,I1,I2,I3,MS,I4,I5,I6,I7)
C
C
      CALL DFILW("RMAG",I)
      IF(I.NE.1) TYPE "CREAD DFILW RMAG=",I
      CALL DFILW("ANG",I)
      IF(I.NE.1) TYPE"CREAD DFILW ANG=",I
      CALL CFILW("RMAG",3,512,I)
      IF(I.NE.1) TYPE "CREAD CFILW RMAG=",I
      CALL CFILW("ANG",3,512,I)
      IF(I.NE.1) TYPE"CREAD CFILW ANG=",I
      CALL OPEN(3,INFILE,1,I)
      IF(I.NE.1)TYPE"CREAD OPEN3=",I
      CALL OPEN(4,"RMAG",3,I)
      IF(I.NE.1) TYPE"CREAD OPEN4=",I
      CALL OPEN(5,"ANG",3,I)
      IF(I.NE.1) TYPE"CREAD OPEN5=",I
C
      DO 1 I=0,255
      CALL RDBLK(3,(I*4),IO,4,IE)
      IF(IE.NE.1) TYPE"CREAD RDBLK=",IE,I
        DO 2 J=1,256
        RMAG(J)=CABS(MA(J))
        IF(RMAG(J).EQ.0.0) GOTO 44
        X=REAL(MA(J))
        Y=AIMAG(MA(J))
        ANG(J)=ATAN2(Y,X)
        GOTO 2
 44     ANG(J)=0.0
  2     CONTINUE
      CALL WRBLK(4,(I*2),IO1,2,IE)
      IF(IE.NE.1) TYPE"CREAD WRBLK4=",IE
      CALL WRBLK(5,(I*2),IO2,2,IE)
      IF(IE.NE.1) TYPE "CREAD WRBLK5=",IE
1        CONTINUE
         CALL RESET
         STOP   Magnitude and phase files created, RMAG, ANG.
         END
```

PROGRAM HOREV

```
C
C          Program HOREV takes the magnitude file from
C     the template image and combines it with the phase
C     file from the image and combines them to make a
C     modified 256x256 complex image file.
C
C
C
C
C
      COMPLEX MA(256)
      DIMENSION RMAG(256),ANG(256)
      COMMON IO1(1024),IO2(512),IO3(512)
      INTEGER INFILE1(7),INFILE2(7),OUTFILE(7)
      EQUIVALENCE(IO1(1),MA(1))
      EQUIVALENCE(IO2(1),RMAG(1))
      EQUIVALENCE(IO3(1),ANG(1))
C
      CALL IOF(3,MAIN,INFILE1,INFILE2,OUTFILE,I,M,I,I,I,I)
C
  2   FORMAT(" ",S13,"Created by HOREV")
C
      CALL DFILW(OUTFILE,IE)
      IF(IE.NE.1) TYPE"HOREV DFILW=",IE
      CALL CFILW(OUTFILE,3,1024,IE)
      IF(IE.NE.1) TYPE"HOREV CFILW=",IE
      CALL OPEN(3,OUTFILE,3,IE)
      IF(IE.NE.1) TYPE"HOREV OPEN3=",IE
      CALL OPEN(4,INFILE1,1,IE)
      IF(IE.NE.1) TYPE"HOREV OPEN4=",IE
      CALL OPEN(5,INFILE2,1,IE)
      IF(IE.NE.1) TYPE"HOREV OPEN5=",IE
C
      DO 4 I=0,255
      CALL RDBLK(4,(I*2),IO2,2,IE)
      IF(IE.NE.1) TYPE"HOREV RDBLK4=",IE,I
      CALL RDBLK(5,(I*2),IO3,2,IE)
      IF(IE.NE.1) TYPE"HOREV RDBLK5=",IE,I
      DO 3 J=1,256
      X=RMAG(J)*COS(ANG(J))
      Y=RMAG(J)*SIN(ANG(J))
      MA(J)=CMPLX(X,Y)
  3   CONTINUE
      CALL WRBLK(3,(I*4),IO1,4,IE)
      IF(IE.NE.1) TYPE"HOREV WRBLK=",IE,I
  4   CONTINUE
      CALL RESET
      WRITE(10,2) OUTFILE(1)
      END
```

PROGRAM FILTER

```
C         Program filter calls a complex 256x256 file
C       and filters it by a function F(ROW,COL) specified
C       by the user.  ROW equals the absolute value of
C       the distance(number of rows) from the 129th row of
C       the array.  COL equals the absolute value of the
C       distance(number of columns) from the 129th column
C       Thus ROW=0, COL=0 specifies the element A(129,129)
C       of array A.  Also ROW=1, COL=0 specifies the ele-
C       ments A(128,129) and A(130,129) of the array A.
C       To change F(ROW,COL) the user must change the
C       function of ROW and COL in this program and
C       recompile and reload the program.
C
C
C
                COMPLEX MA(256)
        COMMON IO(1024)
        INTEGER INFILE(7),OUTFILE(7)
        EQUIVALENCE(IO(1),MA(1))
        CALL IOF(2,MAIN,INFILE,OUTFILE,I1,I2,MS,I3,I4,I5,I6)
        CALL DFILW(OUTFILE,IE)
        IF(IE.NE.1.AND.IE.NE.13)GOTO 12
        CALL CFILW(OUTFILE,3,1024,IE)
        IF(IE.NE.1) GOTO 13
        CALL OPEN(0,OUTFILE,3,IE)
        IF(IE.NE.1)GOTO 14
        CALL OPEN(1,INFILE,1,IE)
        IF(IE.NE.1) GOTO 15
        DO 1 K=0,255
        CALL RDBLK(1,K*4,IO,4,IE)
        IF(IE.NE.1)GOTO 10
        DO 2 J=0,255
        IF(J.GT.128)GOTO 3
        ROW=128-J
        GOTO 4
   3    ROW=J-128
   4    CONTINUE
        IF(K.GT.128)GOTO 5
        COL=128-K
        GOTO 6
   5    COL=K-128
   6    CONTINUE
C*************************************************************
C*************************************************************
C       THE FILTER FUNCTION SHOULD BE CONTAINED IN THIS
C                           SECTION
C*************************************************************
        MA(J+1)=MA(J+1)*SQRT(ROW**2+COL**2)  ; F(ROW,COL)
C***********************************************************
C*************************************************************
   2    CONTINUE
```

```
    CALL WRBLK(0,K*4,IO,4,IE)
       IF(IE.NE.1)GOTO 11
  1    CONTINUE
       WRITE(10,9) OUTFILE(1)
  9    FORMAT(" ",S13,"created by FILTER")
       GOTO 999
 10    STOP"FILTER RDBLK=",IE,K
 11    STOP"FILTER WRBLK=",IE,K
 12    STOP"FILTER DFILW=",IE
 13    STOP"FILTER CFILW=",IE
 14    STOP"FILTER OPEN0=",IE
 15    STOP"FILTER OPEN1=",IE
999    CONTINUE
       END
```

PROGRAM CORRELATE

```
C       Program CORRELATE multiplys two 256x256 com-
C     plex files according to the formula;
C     INFILE1(J,K)*CONJ(INFILE2(J,K))=OUTFILE(J,K).
C     The resulting array is stored in file OUTFILE.
C
C
C
C
      DIMENSION INFILE1(7),INFILE2(7)
      INTEGER OUTFILE(7)
      COMPLEX MA1(256),MA2(256),MA3(256)
      COMMON IO1(1024),IO2(1024),IO3(1024)
      EQUIVALENCE(MA1(1),IO1(1))
      EQUIVALENCE(MA2(1),IO2(1))
      EQUIVALENCE(MA3(1),IO3(1))
C
      CALL IOF(3,MAIN,INFILE1,INFILE2,OUTFILE,I1,MS,I2,
     +I3,I4,I5)
C
   2  FORMAT(" ",S13,"Created by CORRELATE")
C
      CALL DFILW(OUTFILE,IE)
      IF(IE.NE.1) TYPE"CORRELATE DFILW=",IE
      CALL CFILW(OUTFILE,3,1024,IE)
      IF(IE.NE.1)TYPE"CORRELATE CFILW=",IE
      CALL OPEN(1,OUTFILE,2,IE)
      IF(IE.NE.1)TYPE"CORRELATE OPEN1=",IE
      CALL OPEN(2,INFILE1,2,IE)
      IF(IE.NE.1)TYPE"CORRELATE OPEN2=",IE
      CALL OPEN(3,INFILE2,2,IE)
      IF(IE.NE.1)TYPE"CORRELATE OPEN3=",IE
C
      DO 4 I=0,255
      CALL RDBLK(2,(I*4),IO2,4,IE)
      IF(IE.NE.1) TYPE"CORRELATE RDBLK2=",IE,I
      CALL RDBLK(3,(I*4),IO3,4,IE)
      IF(IE.NE.1)TYPE"CORRELATE RDBLK3=",IE,I
      DO 3 J=1,256
      MA1(J)=MA2(J)*CONJG(MA3(J))
   3  CONTINUE
      CALL WRBLK(1,(I*4),IO1,4,IE)
      IF(IE.NE.1)TYPE"CORRELATE WRBLK1=",IE,I
   4  CONTINUE
      WRITE(10,2) OUTFILE(1)
      END
```

PROGRAM RBIG

```
C     Program RMAT takes 9 scenes created by RMAT and
C        combines them into one large 256x256 real scene
C        array.
C
C
C
         DIMENSION IFILE(7)
         CALL IOF(1,MAIN,IFILE,I2,I3,I4,MS,I5,I6,I7,I8)
         CALL DFILW(IFILE,IE)
         IF(IE.NE.1.AND.IE.NE.13)TYPE"BIG DFILW=",IE
         CALL CFILW(IFILE,3,512,IE)
         IF(IE.NE.1)TYPE"BIG CFILW=",IE
          CALL OPEN(1,"MATRIX1",1,IE)
         IF(IE.NE.1)TYPE"BIG OPEN1=",IE
         CALL OPEN(2,"MATRIX2",1,IE)
         IF(IE.NE.1)TYPE"BIG OPEN2=",IE
         CALL OPEN(3,"MATRIX3",1,IE)
         IF(IE.NE.1)TYPE"BIG OPEN3=",IE
         CALL OPEN(4,IFILE,3,IE)
         IF(IE.NE.1)TYPE"BIG OPEN4=",IE
         CALL READ3(1,2,3,4,1)
         CALL RESET
         CALL OPEN(1,"MATRIX4",1,IE)
         IF(IE.NE.1)TYPE"BIG OPEN5=",IE
         CALL OPEN(2,"MATRIX5",1,IE)
         IF(IE.NE.1)TYPE"BIG OPEN6=",IE
         CALL OPEN(3,"MATRIX6",1,IE)
         IF(IE.NE.1)TYPE"BIG OPEN7=",IE
         CALL OPEN(4,IFILE,3,IE)
         IF(IE.NE.1)TYPE"OPENJJ2=",IE
         CALL READ3(1,2,3,4,2)
         CALL RESET
         CALL OPEN(1,"MATRIX7",1,IE)
         IF(IE.NE.1)TYPE"BIG OPEN8=",IE
         CALL OPEN(2,"MATRIX8",1,IE)
         IF(IE.NE.1)TYPE"OPEN9=",IE
         CALL OPEN(3,"MATRIX9",1,IE)
         IF(IE.NE.1)TYPE"BIG OPEN13=",IE
         CALL OPEN(4,IFILE,3,IE)
         IF(IE.NE.1)TYPE"OPEN JJ=3",IE
         CALL READ3(1,2,3,4,3)
         WRITE(10,44) IFILE(1)
  44     FORMAT(" ",S13,"created by RBIG")
         END
```

PROGRAM READ3

```
C     Subroutine READ3 reads in 3 picture arrays created
C   by RMAT and writes them to the 9x9 picture array being
C   created by RBIG.  That is the 9x9 large scene is created
C   one "row" of pictures at a time.
          DIMENSION RA(256)
          INTEGER MA4(512)
          COMMON MA4
          EQUIVALENCE(RA(1),MA4(1))
          REAL MA1(100),MA3(100),MA2(100)
          SUBROUTINE READ3(CH1,CH2,CH3,CH4,JJ)
          INTEGER CH1,CH2,CH3,CH4
          DO 4 J=1,80
          READ BINARY(CH1) MA1
          READ BINARY(CH2) MA2
          READ BINARY(CH3) MA3
          IF(J.LE.5) GOTO 4
          DO 1 I=1,78
    1     RA(I)=MA1(I)
          DO 2 I=79,172
    2     RA(I)=MA2(I-78)
          DO 3 I=173,256
    3     RA(I)=MA3(I-172)
          L=2*(J-5*(JJ-1)-1);          increment
          KK=L+160*(JJ-1)
          CALL WRBLK(CH4,KK,MA4,2,IE)
          IF(IE.NE.1)TYPE"3READ WRBLK=",IE,J,JJ
    4     CONTINUE
          RETURN
          END
```

PROGRAM IOF

```
   SUBROUTINE IOF(N,MAIN,F1,F2,F3,F4,MS,S1,S2,S3,S4)
C
C    Written by Lt. Simmons         31 Aug 1981
C
C    This FORTRAN 5 subroutine will read from the file
C    COM.CM (FCOM.CM in the foreground) the program name,
C    any global switches, and up to four local file names
C    and corresponding switches.
C
C    Calling arguments:
C
C    N is the number of local files and switches to be
C    read from (F)COM.CM.  N must be 1, 2, 3, or 4.
C
C    MAIN is an array for the main program file name.
C
C    F1, F2, F3, and F4 are the four variables to return
C    the local file names.
C
C    MS is a two-word integer array that holds any global
C    switches.
C
C    S1, S2, S3, and S4 are two-word integer arrays that
C    hold the local switches corresponding to F1 through
C    F4 respectively.
C
C    Dimension the arrays.
C
     DIMENSION MAIN(7),MS(2)
     INTEGER F1(7),F2(7),F3(7),F4(7),S1(2)
   +,S2(2),S3(2),S4(2)
C
C    Check the bounds on N.
C
     IF(N.LT.1.OR.N.GT.4)STOP "N out of bounds in IOF"
C
C    Process the data in COM.CM (or FCOM.CM).
C
     CALL GROUND(I)  ;Find out which ground I am in
     IF(I.EQ.0)OPEN 0,"COM.CM"     ;Open ch. 0 to COM.CM
     IF(I.EQ.1)OPEN 0,"FCOM.CM"    ;Open ch. 0 to FCOM.CM
     CALL COMARG(0,MAIN,MS,IER)      ;Read from COM.CM
     IF(IER.NE.1)TYPE" COMARG error:",IER
X    WRITE(10,1)MAIN(1)              ;Type prog. name
X  1 FORMAT(' Program ',S13,'running.')
     CALL COMARG(0,F1,S1,JER)        ;Read from COM.CM
     IF(JER.NE.1)TYPE" COMARG error (F1):",JER
     IF(N.EQ.1)GO TO 2       ;Test N
     CALL COMARG(0,F2,S2,KER)        ;Read from COM.CM
     IF(KER.NE.1)TYPE" COMARG error (F2):",KER
     IF(N.EQ.2)GO TO 2       ;Test N
```

```
  CALL COMARG(0,F3,S3,LER)        ;Read from COM.CM
     IF(LER.NE.1)TYPE" COMARG error (F3):",LER
     IF(N.EQ.3)GO TO 2       ;Test N
     CALL COMARG(0,F4,S4,MER)        ;Read from COM.CM
     IF(MER.NE.1)TYPE" COMARG error (F4):",MER
2    CLOSE 0
     RETURN
     END
```

PROGRAM UNPACK

```
C    PROGRAM UNPACK:  Program unpacks a 64X256 Video
C       picture into a 256Pixel by 256'array located
C       in file UFILE.

        SUBROUTINE UNPACK(NAME,IFILE)
        DIMENSION NAME(7)
        INTEGER IPICT(256),NPICT(1024),A,B
        CALL CFILW (IFILE,2,IER)
        CALL OPEN (1,NAME,0,IER)
        CALL OPEN (2,IFILE,0,IER)
           DO 1 I=0,63
           L=I*4
           CALL RDBLK (1,I,IPICT,1,IER)
                DO 2 J=1,256
                M=J*4
                N=IPICT(J)
                   DO 3 K=1,4
                   A=15.AND.N
                   NPICT(M)=A
                   N=ISHFT(N,-4)
                   M=M-1
3                  CONTINUE
2               CONTINUE
           CALL WRBLK (2,L,NPICT,4,IER)
1          CONTINUE
        CALL RESET
        RETURN
        END
```

PROGRAM REPACK

```
C   PROGRAM REPACK:  Program takes a normalized
C      integer file (NIFILE) and packs it into
C      a file (PICT) usable by Video.

      SUBROUTINE REPACK(IFILE)
      DIMENSION IPICT(1024),NPICT(256)
      INTEGER A,B,C

      CALL CFILW ("PICT",2,IER)
      CALL OPEN (1,IFILE,0,IER)
      CALL OPEN (2,"PICT",0,IER)
         DO 1 I=0,63
         M=I*4
         CALL RDBLK(1,M,IPICT,4,IER)
              DO 2 J=1,1024,4
              L=J
              N=(J+3)/4
              A=0
                 DO 3  K=1,4
                 A=ISHFT(A,4)
                    A=IPICT(L)+A
                 L=L+1
 3               CONTINUE
              NPICT(N)=A
 2            CONTINUE
         CALL WRBLK(2,I,NPICT,1,IER)
 1       CONTINUE
      CALL RESET
      RETURN
      END
```

PROGRAM REDNOSE

```
C    PROGRAM REDNOSE:  This program performs
C       a user specified averaging on previously
C       created video files of  identical
C       images.  Inputs are specified for each
C       iteration by the user.  The final
C       image is output to file VAVG.
C
C
        DIMENSION NAME(7)
        INTEGER FPICT(256),SPICT(256),NPICT(256)
C
        K=0
        ACCEPT" Name of first file? "
        READ(11,15) NAME(1)
15      FORMAT (S13)
        CALL DUNPACK (NAME,"VPICT1")
        TYPE" File is unpacked and in VPICT1."
C
        ACCEPT" Name of second file? "
        READ(11,15) NAME(1)
        CALL DUNPACK (NAME,"VPICT2")
        TYPE" File is upacked and in VPICT2. "
        CALL CFILW ("VAVG",2,IER)
        IF(IER.NE.1)TYPE"File create error.",IER
        CALL OPEN  (1,"VAVG",2,IER)
        IF(IER.NE.1)TYPE"Channel 1 open error 1.",IER
C
        CALL OPEN (2,"VPICT1",2,IER)
        IF(IER.NE.1)TYPE"Channel 2 open error 2.",IER
        CALL OPEN (3,"VPICT2",2,IER)
        IF(IER.NE.1)TYPE"Channel 3 open error 3.",IER
           DO 1 I=0,255
           CALL RDBLK (2,I,FPICT,1,IER)
        IF(IER.NE.1)TYPE"First RDBLK error.",IER
           CALL RDBLK (3,I,SPICT,1,IER)
        IF(IER.NE.1)TYPE"Second RDBLK error.",IER
                DO 2 J=1,256
                NPICT(J)=(FPICT(J)+SPICT(J))/2
2               CONTINUE
           CALL WRBLK (1,I,NPICT,1,IER)
        IF(IER.NE.1)TYPE"First WRBLK error.",IER
1          CONTINUE
        K=K+2
        CALL RESET
C
3       ACCEPT" Do you wish to continue (YES/NO)? "
        READ (11,16) IYN1
16      FORMAT (S1)
        IF (IYN1.EQ.19968) GO TO 9              ;NO TO STOP
        IF (IYN1.NE.22784) GO TO 7              ;NOT YES
        GO TO 8
```

```
7     TYPE" Input error!!! Try again. "
         GO TO 3
C
C
8          ACCEPT" Name of next file? "
           READ(11,15) NAME(1)
           CALL DUNPACK(NAME,"VPICTN")
           TYPE" File is unpacked in VPICTN."
           CALL OPEN (1,"VAVG",2,IER)
           IF(IER.NE.1)TYPE"Second OPEN channel 1 error.",IER
           CALL OPEN (2,"VPICTN",2,IER)
           IF(IER.NE.1)TYPE"2nd Channel 2 error.",IER
           CALL CFILW ("VAVGN",2,IER)
           IF(IER.NE.1)TYPE " 2nd create error.",IER
           CALL OPEN (3,"VAVGN",2,IER)
           IF(IER.NE.1)TYPE" 2nd open channel 3 error.",IER
              DO 10 I=0,255
              CALL RDBLK (1,I,FPICT,1,IER)
           IF(IER.NE.1)TYPE"Third RDBLK error.",IER
              CALL RDBLK (2,I,SPICT,1,IER)
           IF(IER.NE.1)TYPE"4th RDBLK error.",IER
                   DO 17 J=1,256
                   NPICT(J)=(FPICT(J)+SPICT(J))/2
17                 CONTINUE
              CALL WRBLK (3,I,NPICT,1,IER)
           IF(IER.NE.1)TYPE"2nd WRBLK error.",IER
10            CONTINUE
           K=K+1
           CALL RESET
           CALL DFILW("VAVG",IER)
           IF(IER.NE.1)TYPE"VAVG Delete error.",IER
           RENAME "VAVGN","VAVG"
           IF(IER.NE.1)TYPE"Rename failed.",IER
           GO TO 3
C
C
9          CALL DREPACK("VAVG")
12         ACCEPT" Do you wish to rename the averaged file(YES/NO)? "
           READ (11,16) IYN2
           IF(IYN2.EQ.22784) GO TO 13              ;YES
           IF(IYN2.NE.19968) GO TO 14              ;NO
           TYPE" You have averaged ",K," pictures. "
           TYPE" Averaged file is in file PICT. "
           GO TO 18
C
14         TYPE"Input error.  Try again.  "
           GO TO 12
13         ACCEPT"Desired outputfile name? "
           READ (11,15) NAME(1)
           RENAME "PICT",NAME
18         CALL DFILW("VAVG",IER)
           CALL DFILW("VPICT1",IER)
           CALL DFILW("VPICT2",IER)
           CALL DFILW("VPICTN",IER)
           STOP
           END
```

PROGRAM CLEAN

```
C    PROGRAM CLEAN; REVISION 1: Program
C       finishes job started by THRES.

        DIMENSION IO(256),NAME(7)
        INTEGER A,B

        ACCEPT" Name of input file? "
        READ(11,30) NAME(1)
30      FORMAT(S13)
        CALL DUNPACK(NAME,"TPICT")
        CALL OPEN (1,"TPICT",1,IER)
         IF(IER.NE.1)TYPE"OPEN1 ERROR= ",IER
        CALL CFILW ("PUP",2,IER)
         IF(IER.NE.1)TYPE"CREATE ERROR= ",IER
        CALL OPEN (2,"PUP",3,IER)
        IF(IER.NE.1)TYPE"OPEN2 ERROR= ",IER
        ACCEPT"   Threshold value? ",NUM1
           DO 1 I=0,255
           CALL RDBLK (1,I,IO,1,IER)
            IF(IER.NE.1)TYPE"RDBLK ERROR= ",IER
                DO 2 J=1,256
                A=IO(J)
                IF(A.LT.NUM1)GO TO 3
                IO(J)=15
2               CONTINUE
                GO TO 5
3               DO 4 K=1,256
                L=257-K
                B=IO(L)
                IF(B.LT.NUM1)GO TO 5
                IO(L)=15
4               CONTINUE
5          CALL WRBLK (2,I,IO,1,IER)
            IF(IER.NE.1)TYPE"WRBLK ERROR= ",IER
1          CONTINUE
        CALL RESET
        CALL TREPACK("PUP")
        ACCEPT"   Name of output file. "
        READ(11,30) NAME(1)
        RENAME "TPICTN",NAME
        CALL DFILW ("TPICT",IER)
        CALL DFILW ("PUP",IER)
        STOP
        END
```

PROGRAM COMBINE

```
C    PROGRAM COMBINE, VERSION 2: Takes a scene
C       consisting of any single grey level and
C       combines it with a template.

        DIMENSION NAME(7),IO(256),NO(256),LO(256)
        INTEGER A,B
        ACCEPT" Name of template file?` "
        READ(11,30) NAME(1)
30      FORMAT(S13)
        CALL DUNPACK(NAME,"TEMP")
        ACCEPT" Name.of reversed PIMT  file? "
        READ(11,30) NAME(1)
        CALL DUNPACK(NAME,"IMAGE")
        CALL OPEN(1,"TEMP",1,IER)
        CALL OPEN (2,"IMAGE",1,IER)
        CALL CFILW ("NEW",2,IER)
        CALL OPEN (3,"NEW",3,IER)
           DO 1 I=0,255
           CALL RDBLK (1,I,IO,1,IER)
           CALL RDBLK (2,I,NO,1,IER)
                DO 2 J=1,256
                A=IO(J)
                IF(A.EQ.15)GO TO 3
                LO(J)=NO(J)
                GO TO 2
3               LO(J)=A
2               CONTINUE
           CALL WRBLK (3,I,LO,1,IER)
1          CONTINUE
        CALL RESET
        CALL TREPACK("NEW")
        ACCEPT" Name of output file? "
        READ(11,30) NAME(1)
        RENAME "TPICTN",NAME
        CALL DFILW ("TEMP",IER)
        CALL DFILW ("IMAGE",IER)
        CALL DFILW ("NEW",IER)
        STOP
        END
```

PROGRAM MOVE

```
C     PROGRAM MOVE: This program moves a
C        template to a new location by way of a 2-D
C        shift.  It is assumed that the location of
C        the template is known.

         DIMENSION NAME(7),IO(256),NO(256)
         INTEGER A,B,C,D
         ACCEPT" Name of template to be moved? "
         READ(11,30) NAME(1)
30       FORMAT(S13)
         CALL DUNPACK(NAME,"TPICT")
         CALL OPEN (1,"TPICT",1,IER)
           IF(IER.NE.1)TYPE"OPEN1 ERROR= ",IER
         CALL CFILW ("TEM",2,IER)
           IF(IER.NE.1)TYPE"CREATE ERROR= ",IER
         CALL OPEN (2,"TEM",3,IER)
           IF(IER.NE.1)TYPE"OPEN2 ERROR= ",IER
         ACCEPT" Enter old top row number. ",NUM1
         ACCEPT" Old bottom row number? ",NUM2
         ACCEPT" Old right column number? ",NUM3
         ACCEPT" Old left column number? ",NUM4
         ACCEPT" New top row? ",NUM5
         ACCEPT" New left column number? ",NUM6
         ACCEPT" Background grey level? ",NUM7
         A=NUM1-NUM5
         D=NUM3-NUM4
         B=NUM4-NUM6
         C=NUM2-NUM1
         NBR=NUM5+C
         NRC=NUM6+D
         DO 1 I=0,NUM5
            DO 2 J=1,256
            NO(J)=NUM7
2           CONTINUE
         CALL WRBLK (2,I,NO,1,IER)
          IF(IER.NE.1)TYPE"WRBLK1 ERROR= ",IER
1        CONTINUE
         DO 3 I=NUM1,NUM2
         CALL RDBLK (1,I,IO,1,IER)
          IF(IER.NE.1)TYPE"RDBLK ERROR= ",IER
         M=I-A
            DO 4 J=1,NUM6
            NO(J)=NUM7
4           CONTINUE
            DO 5 J=NRC,256
            NO(J)=NUM7
5           CONTINUE
            DO 6 J=NUM4,NUM3
            L=J-B
            NO(L)=IO(J)
6           CONTINUE
```

```
     CALL WRBLK (2,M,NO,1,IER)
         IF(IER.NE.1)TYPE"WRBLK2 ERROR= ",IER
3       CONTINUE
        DO 7 I=NBR,255
           DO 8 J=1,256
           NO(J)=NUM7
8          CONTINUE
        CALL WRBLK (2,I,NO,1,IER)
         IF(IER.NE.1)TYPE"WRBLK3 ERROR= ",IER
7       CONTINUE
        CALL RESET
        CALL TREPACK("TEM")
        CALL DFILW("TEM",IER)
        CALL DFILW("TPICT",IER)
        ACCEPT" Desired name for output file? "
        READ(11,30) NAME(1)
        RENAME "TPICTN",NAME
        STOP
        END
```

PROGRAM EXAMINE

```
C    PROGRAM EXAM:  This program asks for a
C       threshold value.  If the pixel is less than
C       the specified threshold, it is put in
C       the new image.  If not, the pixel is set
C       to bright (grey level 15).

        DIMENSION NAME(7),IO(256),NO(256)
        INTEGER A,B
        ACCEPT"Name of image to be examined? "
        READ(11,30)
30      FORMAT(S13)
        CALL DUNPACK(NAME,"TEMP")
        CALL OPEN (1,"TEMP",1,IER)
         IF(IER.NE.1)TYPE"OPEN1 ERROR= ",IER
        CALL CFILW ("NEW",2,IER)
         IF(IER.NE.1)TYPE"CREATE ERROR= ",IER
        CALL OPEN (2,"NEW",3,IER)
         IF(IER.NE.1)TYPE"OPEN2 ERROR= ",IER
        ACCEPT" Number of threshold value? ",NUM1
           DO 1 I=0,255
           CALL RDBLK (1,I,IO,1,IER)
            IF(IER.NE.1)TYPE"RDBLK ERROR= ",IER
                DO 2 J=1,256
                A=IO(J)
                IF(A.LT.NUM1)GO TO 3
                NO(J)=15
                GO TO 2
3               NO(J)=A
2               CONTINUE
           CALL WRBLK (2,I,NO,1,IER)
            IF(IER.NE.1)TYPE"WRBLK ERROR= ",IER
1       CONTINUE
        CALL RESET
        CALL TREPACK("NEW")
        ACCEPT"  Name of output? "
        READ(11,30) NAME(1)
        RENAME "TPICTN",NAME
        CALL DFILW ("TEMP",IER)
        CALL DFILW ("NEW",IER)
        STOP
        END
```

Vita

Darryl Stroup was born 5 August 1953. He graduated from Orion High School, Orion Illinois in 1971. He attended Blackhawk College until he enlisted in the U.S. Air Force on 7 August 1972. He was a Medical Administrative Specialist at Peterson Field CO from December 1972 to October 1974. He attended the University of Utah under the Airman's Education and Commissioning Program (AECP) from October 1974 to June 1977, graduating Cum Laude with a Bachelor Science in Electrical Engineering. He was commissioned from Officer Training School in September 1977. He was an Instrumentation Engineer at Edwards AFB, CA from September 1977 until June 1980 at which time he was assigned to the U.S. Air Force Institute of Technology.

Thomas Dorsey was born 5 January, 1953. He attended Quabbin Regional Jr.-Sr. High School in Barre, Massachusetts from 1967-1971. He attended Worcester Polytechnic Institute in Worcester, Massachusetts from 1972 to 1974. He spent 3 years in the U.S. Army. He completed his B.S.E.E. at the University of Lowell, Massachusetts. He was commissioned in the U.S. Air Force through ROTC in June 1980. He attended the School of Engineering, Air Force Institute of Technology from June 1980 until December 1981.

AD-A115 563 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 20/6
SCENE ANALYSIS - APPLICATION OF TWO-DIMENSIONAL NONLINEAR FILTE--ETC(U)
DEC 81 D R STROUP, T D DORSEY
UNCLASSIFIED AFIT/GE/GEO/EE/81D-57 NL

2 of 2

END
DATE
FILMED
7 82
DTIC

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GE/GEO/EE/81D-57	2. GOVT ACCESSION NO. AD-A115563	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) SCENE ANALYSIS-APPLICATION OF TWO-DIMENSIONAL NONLINEAR FILTERING FOR TARGET ENHANCEMENT AND RECOGNITION		5. TYPE OF REPORT & PERIOD COVERED MS Thesis 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Darryl R. Stroup and Thomas D. Dorsey		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT/EN) Wright-Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE December 1981 13. NUMBER OF PAGES 95
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

15 APR 1982

APPROVED FOR PUBLIC RELEASE AFR 190-17.

18. SUPPLEMENTARY NOTES

Approved for public release; distribution unlimited.

~~FREDRIC C. LYNCH, Major, USAF~~ Dean for Research and Professional Development

~~Director of Public Affairs~~

Air Force Institute of Technology (ATC)
Wright-Patterson AFB, OH 45433

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

PATTERN RECOGNITION
NONLINEAR FILTERING
TARGET DETECTION
IMAGE PROCESSING
INFRARED

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A nonlinear scene analysis algorithm is studied for complex scenes containing realistic targets and background clutter. Infrared and visible spectrum light images are processed. The algorithm combines the Fourier transform phase array of a scene with the Fourier transform magnitude of a template of a template to create a new image. Clutter reduction ability and target recognition capability are examined in detail.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)